A Space-Time Meshfree Collocation Method for Coupled Problems on Irregularly-Shaped Domains

Von der
Fakultät Architektur, Bauingenieurwesen und Umweltwissenschaften
der Technischen Universität Carolo-Wilhelmina
zu Braunschweig

zur Erlangung des Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte

Dissertation

von

Hennadiy Netuzhylov, MSc

geboren am 20.11.1978
aus Lviv, Ukraine

Eingereicht am	12. Juni 2008
Disputation am	10. November 2008
Berichterstatter	Prof. Dr.-Ing. Andreas Zilian
	Prof. Dr. rer. nat. Thomas Sonar

2008

ISBN 978-3-00-026744-4

Herausgeber: Prof. Dr.-Ing. Andreas Zilian

Dedicated to my parents.

Kurzfassung

In der vorliegenden Arbeit wird eine neuartige gitterfreie Raum-Zeit-Kollokationsmethode (*engl.* STMCM) zur Lösung von Systemen partieller und gewöhnlicher Differentialgleichungen durch eine konsistente Diskretisierung in Raum und Zeit als Alternative zu den etablierten netzbasierten Verfahren vorgeschlagen. Die STMCM gehört zur Klasse der tatsächlich gitterfreien Methoden, die nur mit Punktwolken ohne a priori Netzkonnektivität arbeiten und kein Diskretisierungsnetz benötigen. Das Verfahren basiert auf der Interpolating Moving Least Squares Methode, die eine vereinfachte Erfüllung der Randbedingungen durch die von den Kernfunktionen erfüllte Kronecker-Delta-Eigenschaft ermöglicht, was beim größten Teil anderer netzfreier Verfahren nicht der Fall ist. Ein Regularisierungsverfahren zur Bewältigung des beim Aufbau der Kernfunktionen auftretenden Singularitätsproblems, sowie zur Berechnung aller benötigten Ableitungen der Kernfunktionen wird dargelegt. Ziel ist es dabei, eine Methode zu entwickeln, die die Einfachheit der Verfahren zur Lösung partieller Differentialgleichungen in starker Form mit den Vorteilen der gitterfreien Verfahren, insbesondere mit Blick auf gekoppelte Probleme des Ingenieurwesens mit sich bewegenden Grenzflächen, verknüpft.

Die vorgeschlagene Methode wird zunächst zur Lösung linearer und nichtlinearer partieller sowie gewöhnlicher Differentialgleichungen angewendet. Dabei werden deren Konvergenz- und Genauigkeitseigenschaften untersucht. Die implizite Rekonstruktion der Gebiete mit komplizierten Rändern als Abbildungsstrategie zur Punktwolken-Streuung wird durch die Interpolation von Punktwolkendaten in zwei und drei Raumdimensionen demonstriert. Anhand der Modelle zur Simulation von Biofilm- und Tumor-Wachstumsprozessen werden Anwendungsbeispiele aus dem Bereich der Umweltwissenschaften und der Medizintechnik dargestellt.

Stichwörter Raum-Zeit · gitterfrei · netzfrei · Kollokation · Interpolating Moving Least Squares · starke Form · Biofilm · Tumor · Differentialgleichungen · Punktwolken · Streudaten.

Abstract

In this thesis an innovative Space-Time Meshfree Collocation Method (STMCM) for solving systems of nonlinear ordinary and partial differential equations by a consistent discretization in both space and time is proposed as an alternative to established mesh-based methods. The STMCM belongs to the class of truly meshfree methods, i.e. the methods which do not have any underlying mesh, but work on a set of nodes only without an a priori node-to-node connectivity. The STMCM is constructed using the Interpolating Moving Least Squares technique, allowing a simplified implementation of boundary conditions due to fulfilment of the Kronecker delta property by the kernel functions, which is not the case for the major part of other meshfree methods. A regularization technique to overcome the singularity-by-construction problem and compute all necessary derivatives of the kernel functions is presented. The goal is to design a method that combines the simplicity and straightforwardness of the strong-form computational techniques with the advantages of meshfree methods over the classical ones, especially for coupled engineering problems involving moving interfaces.

The proposed STMCM is applied to linear and nonlinear partial and ordinary differential equations of different types and its accuracy and convergence properties are studied. The power of the technique is demonstrated by implicit reconstruction of domains with complex boundaries via interpolation of point cloud data in two and three space dimensions as a ‘mapping’ strategy for distribution of computational points within such domains. Applications from the fields of environmental and medical engineering are presented by means of a mathematical model for simulating a biofilm growth and a nonlinear model of tumour growth processes.

Keywords space-time · meshfree · meshless · collocation · Interpolating Moving Least Squares · strong form · biofilm · tumour · differential equations · point clouds · scattered data.

Contents

"You are never given a wish without also being given the power to make it come true. You may have to work for it, however."

-- Richard Bach

Preface

Braunschweig, March 2008

My interest in meshfree methods initiated during my work as a research assistant in the Institute of Computational Mathematics and developed further while working as a scientific assistant at the MSc Programme "Computational Sciences in Engineering" at the Technische Universität Braunschweig in Germany. It is my great pleasure to thank at this stage those people without whom this thesis would not have been possible.

First of all I would like to express my gratitude to my doctoral thesis supervisor Prof. Dr. Andreas Zilian for numerous thought-provoking discussions, his continuous commitment to and interest in the topic, for his encouragement, inspiration, and support throughout the whole period, as well as for giving me the opportunity to gather valuable teaching experiences in various fields.

I would also like to thank my scientific supervisor Prof. Dr. Thomas Sonar for familiarizing me with meshfree methods, for not setting limits to my research activities but rather transmitting his remarkable passion for mathematics and numerical methods, for many discussions, his support and highly-competent advice.

I thank my dear colleagues for all their joy of life and permanent support, for the one-of-a-kind working atmosphere, for teaching me German idioms every now and then, and for many non-scientific activities.

Further thanks go to my family in Ukraine for their love, for their belief in me, for constant encouragement and endless support.

Last, but certainly not least I would like to thank my dear friends I am fortunate to have for being the part of my life I do not want to miss.

Hennadiy Netuzhylov

"Computers are useless. They can only give you answers."
-- Pablo Picasso

"A positive attitude may not solve all your problems, but it will annoy enough people to make it worth the effort. "
-- Helm Albreit

CHAPTER 1

Introduction

The investigation of various engineering systems and natural phenomena is based upon mathematical models that consist of systems of nonlinear partial differential equations (PDEs). Nowadays it is nearly impossible to distinguish between engineering and computational sciences, as the numerical solution of PDEs has become an indispensable part of the engineering practice when modelling e.g. propagation of electromagnetic waves, fluid flows, diffusion of chemical processes, dynamics of structures, acoustics, geophysics, biomechanics, to name just a few.

In addition to often costly, scale-dependent or even impossible experiments (e.g. crash analysis, earthquakes, tsunamis, fire distribution, nuclear technology, dynamics of climate, etc.), numerical simulations are carried out to investigate the physical phenomena using certain computational techniques and approximate solutions. Reliability, robustness, stability, accuracy, and convergence of these approximations are essential issues of engineering research activities.

The classical numerical methods, also known as the *'big three technologies'* have appeared in literature in consecutive decades, i.e. finite differences in the 1950s, finite elements in the 1960s, and spectral methods in the 1970s. Meshfree methods (in literature also called 'meshless', 'gridless', or 'particle' methods), which are the subject of this thesis, have their roots back in the 1970s and started to appear in literature in the 1980s.

1.1 Motivation

Why did the meshfree methods become a hot discussion and investigation topic, as we shell see in the next section? And what makes some people argue that these methods are on their way to become the *'fourth big technology'* in the numerical analysis?

To answer these questions let us begin with looking at a standard procedure when applying numerical methods to the mathematical modelling of engineering problems, Figure 1.1 demonstrates a general approach.

Making some assumptions and simplifications to a real physical or engineering problem, a mathematical model is built which is then discretized using appropriate numerical methods, and a numerical solution is obtained. Using certain adaptivity criteria, the accuracy of the numerical solution can be increased, and modifying the assumptions made, the model applicability can be improved. On interpreting the results, one obtains an engineering solution to the problem that describes physical phenomena under consideration.

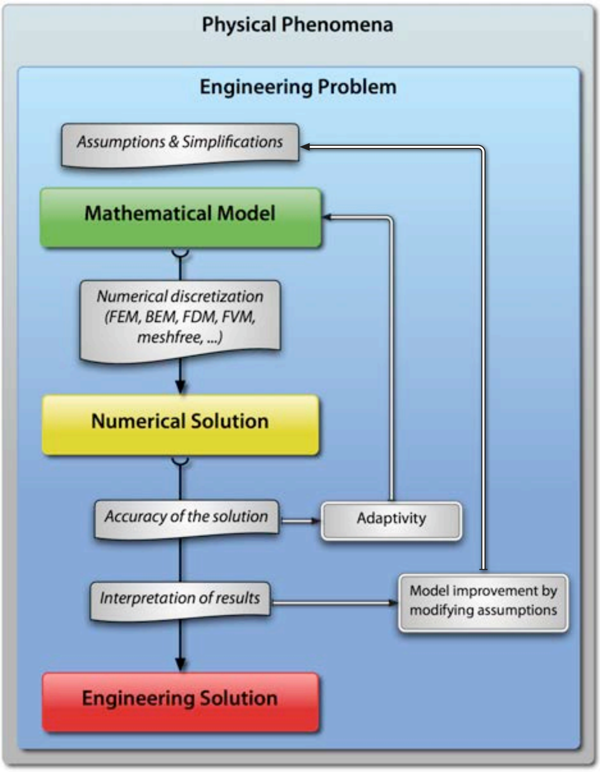


Figure 1.1 | A general process of mathematical modelling.

In all classical mesh-based numerical methods some sort of an *a priori* node-to-node connectivity is used during the step of numerical discretization of a mathematical model. A *mesh* used in finite difference method (FDM) is often called *grid*, in finite volume method (FVM) it consists of *volumes* or *cells*, and in finite element method (FEM) – of *elements*. Although these different terms do have certain physical meanings, they all can be entitled as *mesh* since in all these cases a specific relationship between the nodes must be established *in advance*.

Using this properly predefined mesh, a system of governing PDEs is then discretized by a set of algebraic equations.

The most time consuming, repetitious, and still not fully automated part in the numerical simulation using the mesh-based methods, especially for 3d, is the mesh generation. Moreover, the mesh-based methods like finite elements suffer from accuracy issues when dealing with problems with evolving boundaries, free-surfaces, extremely large deformations, e.g. metal forming, impact extrusion, molding, fracture and failure analysis, e.g. propagation of cracks, etc. due to distortion of the elements as well as from a high computational cost because of the need to repeatedly re-mesh the changing domains. These downsides of the classical computational methods motivated the development of new ones particularly well-suited for problems with moving boundaries. An overview will be given in the next section.

The main difference between the mesh*free* methods and the mesh-*based* ones, as the name already implies, is the absence of the mesh in the former ones. The discretization is maintained on a set of points (also called *nodes, particles*) only. Based on this set of points the meshfree kernel functions can be constructed, and the point-to-point connectivity, which may change during the computation, is established on the fly. In further sections we will present a fast search technique for finding neighbouring points during the computation. In other words, in meshfree methods a discretization of governing PDEs by a system of algebraic equations is maintained without use of a predefined mesh, but on a set of nodes only.

The h-adaptivity can be implemented in a much simpler way compared to the mesh-based methods, since nodes can be added (or removed) as needed without the costly re-meshing procedure – only a small part of neighbouring kernel functions must be recomputed due to their local character. The p-adaptivity is also simpler (or at least just as simple) to perform in meshfree methods than in the mesh-based ones. As we shall see in further sections, meshfree methods are very well suited to be parallelized due to local functions used for constructing the discretization of the problem. Moreover, the approximation functions fulfill the continuity requirements by construction, and no post-processing is needed to determine the derivatives of the unknowns.

One of the disadvantages of the meshfree methods is their higher computational cost. These methods, when applied in a weak or 'variational' form of governing equations, may be up to ten to fifty(!) times more computationally expensive than their mesh-based counterparts and the number of Gaussian quadrature nodes for the numerical integration is much higher [68, 93], since an accurate and careful evaluation of integrals is needed.

Another big issue with almost all of the meshfree methods mentioned in Section 1.2 on the next page, in contrast to the mesh-based methods like FEM, is the lack of the Kronecker delta property. This issue led necessarily to the problems with the imposition of essential boundary conditions, and a number of publications address this problem and propose different approaches to overcome it. The approach presented in this thesis, as we shall see in later sections, eliminates the problem, i.e. our kernel functions do enjoy the Kronecker delta property.

In this thesis a strong-form formulation for differential equations is used and its straightforwardness is combined with the advantages of the meshfree methods described above.

The key features of the proposed approach can be summarized as:

(i) no need to generate a mesh,
(ii) no need to evaluate integrals,
(iii) simplified application of boundary conditions,
(iv) applicability to irregularly-shaped domains.

1.2 Historical development

The description of problems of mathematical physics by means of partial differential equations goes back to the work of Courant, Friedrichs and Lewy [42] in 1928, when they defined a condition on the time step for certain schemes to be convergent. A huge amount of books dealing with numerical analysis and solution of partial differential equations using the classical 'big three' methods are available. Some of the classical and recommended ones in this area are the books by Ames [8], Morton [144], Strauss [188], Trefethen [194], Quarteroni and Valli [170]. There are also works available that consider only equations and problems of a certain type, e.g. Hackbusch [79] and Young [208], dealing with the elliptic PDEs; or Le Veque [111] and Meister [140], which focus on equations of hyperbolic type. We will give a classification of PDEs in later sections.

As already mentioned before, for problems with evolving and moving boundaries and large deformations the classical mesh-based methods are not the best choice and a number of methods specifically designed for such class of problems were developed. Hirt et al. presented in 1974 an Arbitrary-Lagrange-Eulerian (ALE) method [87], where the Lagrangian description of solids is combined with an Eulerian description of fluid particles. In an Immersed Boundary Method (IBM), presented by Peskin in 1972 for description of heart flow [166, 167], the bodies boundaries are directly immersed into a fixed Cartesian grid. A similar method by Le Veque [112] introduced in 1994, an Immersed Interface Method (IIM), is a modification of the IBM of second order accuracy and in the same year a fictitious domain method related to the IBM was presented by Glowinski et al. [72], in which immersed bodies are coupled to fluid by Lagrange multipliers.

The first numerical method employing the *meshfree* idea (not taking into account the first generalizations of finite differences, which will be discussed later) seems to be the Smoothed Particle Hydrodynamics (SPH) method [137] introduced by Lucy in 1977, which was applied to the astrophysical problems without boundaries like supernova, exploding stars and dust clouds [143, 142] and later on to problems of computational fluid dynamics. Most of the meshfree methods are Lagrangian type methods, but there are also meshfree methods in Eulerian or ALE description [68]. The early SPH-methods experienced numerical instabilities and inconsistency issues and many corrected SPH-methods have been developed, see e.g. the overviews [14, 172]. A number of methods has been developed based on weak form of underlying PDEs, while the original SPH and several corrected approaches were based on strong form of equations.

Further developments of the meshfree methods were most probably motivated by investigation of crack propagation problems and the need of computationally expensive remeshing. Nayroles et al. [146] used the Moving Least Squares in a Galerkin weak formulation in 1992 and called it the Diffuse Element Method (DEM). The Element-Free Galerkin (EFG) method proposed by Belytschko [22, 20] was developed in 1994 and applied to crack propagation problems. This method, as the majority of the meshfree methods, is based on the Moving Least Squares (MLS) approximation technique which will be discussed in detail in further sections. Desimone et al. applied the MLS technique in 1998 to model the Stokes equations [44]. Some recent papers analyze the propagation of cracks in three dimensions via enrichment by a weight function, see e.g. Duflot [48] published in 2006. An overview of the "approximation power" of moving least squares published by Levin in 1998 [114] contributed to the resonance on meshfree methods in the community of Computational Mechanics.

A similar method to EFG, the Reproducing-Kernel Particle Method (RKPM), was developed by Liu in 1995 [127, 131] based on wavelets rather than on curve-fitting, and it was applied by Liu to the problems of structural dynamics in [130]. Overviews and application of the RKPM-method [129, 21] as well as studies on multiscale particle kernel methods [128] are available. The similarities of the RKPM- and MLS-methods were investigated by the authors in [132] and systemized by Fies in 2005 in [67, 68]. Further developments of the RKPM with nodal interpolation properties were made in 2003 by Chen [31].

Some similarities between the finite elements and the meshfree methods were discovered and formulated by Babuška in 1997 in the Partition of Unity Finite Element Method (PUFEM) [16, 141] and in the meshless finite element method [95] by Idelsohn in 2002. Krysl and Belytschko [107] presented an efficient partition of unity method for meshless unstructured methods of linear-precision in 2000. This class of methods (EFG, PUFEM, RKPM, DEM) is consistent and stable in the forms proposed. At the same time it is significantly more computationally expensive (up to ten times!), since a very accurate integration is required. Oñate et al. have published a series of works on finite point method applied to advection-diffusion fluid problems [157, 158, 156, 155, 160], applied it to elasticity problems [159], and together with Löhner and Idelsohn to compressible flow problems [134] in 2002. The hp-cloud method [46, 120], proposed by Duarte, Oden and Liszka in 1996 introduces additional unknowns, increasing herewith the order of consistency of the weak-form formulation by an *extrinsic basis*. This idea was inherited in the standard eXtended finite elements (XFEM) for crack propagation problems. Recently, in 2006, a new method for treatment of arbitrary discontinuities in a variational context was proposed by Fries [69]. In contrast to XFEM, the *intrinsic* XFEM method introduces no additional unknowns.

The Meshless Local Boundary Integral Equation (MLBIE) [14, 215] and the Meshless Local Petrov-Galerkin (MLPG) [12, 13] methods proposed by the group of Atluri in 1998 use the Moving Least Squares to approximate the solution variables. Whereas the MLBIE method uses a local boundary integral equation formulation, the MLPG method deploys a local symmetric weak form. The main difference to the methods mentioned above is that the two latter methods use local weak formulation with overlapping subdomains rather than the global weak formulation. The two methods are, however, *truly meshfree*, since there is no need of a finite element or boundary element mesh, neither for interpolation of the solution, nor for the integration of the weak form. The integration of the weak form is carried out in these approaches using local subdomains. In 1999 Atluri et al. apply the MLPG method to the analysis of thin beams [11] and in 2005 Li applies it to the analysis of thick and thin plates [115]. In these publications (and in some other publications as well) the authors erroneously use the *interpolants* terminology, whereas the *approximation* should have been utilized, since the kernel functions do not satisfy the Kronecker delta property.

Although the majority of researchers focused on the development of meshfree methods based on the *weak* form of governing equations, there have also been approches to solve the equations in the *strong* form. The first meshfree methods in the strong form were the generalizations of finite differences. Jensen published investigations on finite differences on variable grids [97] in 1972, followed by the works of Perrone et al. [165] in 1975. Heinrich presented the finite difference method on irregular networks [85] in 1987, where a generalized approach to second order elliptic problems was investigated. In the same time period Liszka published works on the finite difference method at arbitrary irregular grids [121, 119]. Schönauer derives the formulae of arbitrary consistency order on an unstructured grid [178] in 1998, introducing a procedure to select specific data points to lower the condition number of the linear system. This Finite Difference Element method (FDEM) is, however, not a truly meshfree method, since it requires a priori information on node-to-node connectivity.

In 1995 Schaback introduces the compactly supported radial basis functions (RBFs) [176, 177] and these are also used in the same year by Wendland in [201]. Fasshauer applies the RBFs to solve PDEs by collocations [56] in 1997. Most of the RBFs have a non-compact support, and a global polynomial is added to the basis in order to provide completeness. Vila introduced in 1999 the renormalized meshless derivative (RMD) method for conservation laws [197] which are, however, only 0-th order consistent. Franke and Schaback provide a convergence analysis of the symmetric collocation method by RBFs in 1998 [66] and establish error bounds [65], which require the solution of the PDEs to be very smooth.

1.3 Current state of research

In the period from 2000 to the time of writing this thesis, a series of publications on different variations of collocation methods appeared in the literature. Collocation methods are a subclass of weighted residuals methods with the test functions chosen as Dirac delta distributions, leading immediately to the strong form of governing equations to be evaluated at a set of discrete nodes – *collocation nodes* – in the computational domain. The term *collocation method* is also referred to as the *strong-form method* in the literature, and these two terms will be used equally in this thesis.

Two major advantages of the collocation methods are:

(i) Since no integration is required, the construction of the final system of equations is very efficient.

(ii) The kernel functions must be evaluated at the collocation nodes only in contrast to other methods, where the evaluation at Gauss integration points is required.

Moreover, the number of Gauss integration points in meshfree methods is also significantly larger than in the mesh-based conventional methods like FEM. The drawback of the collocation approach is quite obvious: higher order derivatives of the kernel functions must be computed and evaluated at the collocation nodes, whereas the highest order depends on the order of underlying PDEs.

A rather popular non-symmetric multiquadrics (MQ) collocation method was presented in 1990 by Kansa in [101, 102], where he applied it to solutions of various types of PDEs by radial basis functions. In the second paper the author compares his approach to finite differences and reports its higher accuracy when applied to smooth elliptic problems in fluid dynamics. In 2000 Wagner and Liu address the problem of application of essential boundary conditions in meshfree methods and present a corrected collocation method [199] and Chen and Wang address the contact problems and the imposition of boundary conditions for meshfree methods in [33]. In the same year Aluru proposed a point collocation method [7] based on RPKM, followed by Zhang et al. [212, 213] with their Least-Squares Collocation Method (LSCM). In 2000 Belytchko et al. publish results on stability analysis of meshless particle methods [19]. Fürst and Sonar make preliminary investigations on meshless collocation approximations of conservation laws based on John's positive schemes theory for parabolic equations in 2001 [70]. Kim et al. combine in [103] the RPKM with MLS in the framework of Point Collocation method, and Armentano and Duran publish their studies on error estimates for moving least squares approximations [9].

Meshfree methods as a counterpart to the Boundary Element Method (BEM) have also been developed, see e.g. the boundary knot method by Chen [35, 34, 36] and it is applied to 2d and 3d Helmholtz and convection-diffusion problems with complicated geometries in [89]. In 2002 Fasshauer presents the approximate MLS approximation [57] for time-dependent PDEs, and matrix-free multilevel MLS method [58], and Behrens, Iske et al. present an adaptive meshfree method for nonlinear transport equations in [18]. In 2003 Sridar et al. publish a paper on an upwind finite difference scheme [186] which can be applied in a meshfree context.

Li and Liu publish an overview article [116] on meshfree methods and their application in 2002. The workgroup of Liu presents a Radial Point Interpolation Collocation method (RPICM) in 2003 applying it for the solution of a two-phase flow through the porous media [133]. In the same year a meshless collocation method based on local optimal point interpolation is published by Zuppa [217]. An idea to combine the weak form with the strong one to solve the incompressible flow problems is presented by Liu in [125].

In 2004 Ho et al. present a meshless collocation method based on RBFs and wavelets [88], Ling et al. publish results on meshless collocation techniques [118], and Lee applies the meshfree point collocation method for elasticity and crack propagation problems [113]. A nice review by Huerta et al. on meshfree methods appear in 2004 as a chapter in Encyclopedia of Computational Mechanics [93] covering the approximation techniques by meshfree methods, discretization of PDEs as well as RBFs and the question of blending meshfree methods and finite elements. In 2004 GR Liu and his workgroup also propose a meshfree method based on a weak-strong form (MWS) [125] and apply it to the incompressible flow problems.

But also combination of classical methods with meshfree ones is an area of interest of some researchers, e.g. in 2004 the authors in[104] present the solution of Euler equations embedding a Cartesian grid, allowing a multigrid acceleration, with boundary condition treatment by a meshfree method. In 2005 Cheng and Cabral suggest a direct solution of ill-posed boundary value problems by RBFs [37]. In 2005 Pauly et al. presented the fascinating possibilities of the application of meshfree methods animating the fracture of solids in [164]. Quinlan et al. describe truncation errors [171] in meshfree particle methods in 2005. A guide to an efficient employment of RBFs, published in 2005 by Wendland [202], disscusses the ill-conditioning as well as the correct choice of the basis functions, and Young et al. propose a non-singular boundary-type meshfree methods [209] to solve the potential problems.

In 2006 the workgroup of Liu presents a regularized strong-form meshfree method for adaptive analysis [124], and in the same year they present a stabilized Least-Squares Radial Point Collocation Method (LS-RPCM) and apply it to adaptive analysis as well as the RPCM based on finite differences applied to mechanics problems [126]. Some convergence analysis of the meshfree point collocation method is carried out by Wan and Liu in [200] in 2006, and in the same year von Nida applies the meshfree methods to the dynamics of solids [198] and Liew et al. propose a boundary element-free method (BEFM), which combines the boundary integral equation (BIE) method with the improved moving least squares[1] [117], and apply it to two-dimensional elasticity problems. Zhou and Li apply the RKPM and solve phase field equations in [214] one and two dimensions in 2006.

Gu and GR Liu summarize the meshfree collocation techniques for convection dominated problems in 2006 in the review paper [77] to overcome instability issues. We will refer to this paper in further sections which deal with hyperbolic PDEs. In 2006 GR Liu et al. apply their Least Squares Point Collocation Method (LS-RPCM) for adaptive analysis. Chen et al. combine in [32] the enhanced radial basis collocation method with their reproducing kernel approach. Error estimates of meshfree collocation methods when applied to dynamical systems are outlined by Giesl and Wendland in [71] and the stability of meshfree symmetric collocations for boundary value problems is investigated by Wendland [204] in 2007, and in the same year a weighted radial basis collocation is presented in [90] by Hu for boundary value problems. An interesting alternative to Lagrange multipiers as a strategy to apply the boundary conditions is proposed in the recent article of Tian and Yagawa in 2007 [191], where they "glue" the non-matching grids performing the meshfree interpolation. Several books on meshfree methods have been published recently, e.g. Liu [122, 123], Fasshauer [59], the latter inspired by another well-written book by Trefethen on spectral methods [193].

1.4 Classification of meshfree methods

We have outlined the historical development and the genesis of the meshfree methods in Section 1.2 on page 4 and listed the methods which appeared in recent years, especially from the strong-form point of view, in Section 1.3 on page 6. But are all of these numerous meshfree methods *really free of a mesh*? And which *requirements* must a numerical method fulfill to be called meshfree?

---

[1] The notation of an *Improved* moving least squares method (IMLS) used in [117] coincides with the abbreviation we use for the Interpolating Moving Least Squares.

A *necessary* requirement on a numerical method to be meshfree is quite obvious: no predefined mesh is demanded. An *ideal* requirement would be: no mesh is needed during the whole solution procedure for a problem of a given arbitrary geometry subjected to all kinds of boundary conditions.

All existing meshfree methods may be classified w.r.t. the discretization approach of the governing PDEs, i.e. whether the weak form, the strong form, or a combination of the two is being considered, and most of them are not ideal in the sense described above, they fail to fulfill the ideal requirements on a method to be meshfree and may be categorized into the following groups:

- Numerical methods in which a background mesh (background cells) are required to evaluate the integrals appearing in the system matrix due to the use of the weak form of governing equations. The meshfree methods belonging to this class, e.g. EFG, hp-clouds, PUFEM, etc. are *not truly meshfree*.
- Methods in which background cells are required locally to evaluate the integrals. These methods, e.g. MLPG, are *essentially meshfree*, since a local mesh can be created automatically and it must not be predefined. This construction can be accomplished in a simpler way than constructing a background mesh for the whole domain of interest.
- Methods in which no mesh at all is required. These methods are *truly meshfree*. Finite difference methods on irregular sets of nodes and collocation methods as well as the approach proposed in this thesis belong to this class. The most significant advantage of these methods is no need of integration and, therefore, the straightforwardness and simplicity of implementation as well as lower computational cost.
- Particle methods, e.g. SPH method, which require a predefinition of volumes and/or masses, and consequently *some sort of a mesh is needed*. The conservation of mass in such methods is a big issue.

1.5 Outline of the thesis

Chapter 2 on page 11 begins with a deduction of the moving least squares approximation and interpolation method, which is the core of the proposed Space-Time Meshfree Collocation approach, and compares it with other least square methods. The choice of basis and weight functions is discussed. Subsequently, a rigorous mathematical derivation of a regularization technique to overcome the singularity-by-construction problem and to obtain the nodal values of the kernel functions together with their derivatives is presented. Furthermore, the implementation of boundary and initial conditions for the solution of ordinary and partial differential equations as well as important properties of the kernel functions are described.

The fundamental idea of the Space-Time Meshfree Collocation method (STMCM) is stated in Chapter 3 on page 35. It explains the discretization by Interpolating Moving Least

Squares and outlines the importance of point distribution as well as effective data management techniques. Furthermore, this chapter gives guidelines on choosing the parameters of the STMCM and a brief summary of a typical workflow of mathematical modelling by STMCM.

Chapter 4 on page 47 deals with interpolation of scattered data, where a multilevel surface reconstruction as well as interpolation of irregular (two- and three-dimensional) point cloud data by implicit surfaces are demonstrated. Herewith, a mapping technique for the distribution of computational points within irregularly-shaped domains with complex boundaries is presented.

In Chapter 5 on page 63 interpolating properties of the proposed approach are studied and the method is applied to highly-oscillating functions and their derivatives. Error indicators used to study the accuracy and convergence properties of the STMCM are given.

The STMCM is then applied to various classes of equations in Chapter 6 on page 73. A classification of equations occurring in the numerical simulation of engineering problems according to their types is given, and numerical studies of the accuracy and convergence properties of the method applied to linear and nonlinear systems of ordinary and partial differential equations are performed.

An application of the method to the modelling of biofilm and tumour growth processes is the topic of Chapter 7 on page 131. After a brief overview of what the biofilms are and emphasis of their importance in engineering and everyday life a mathematical model for biofilm growth is presented and some validation examples are discussed. Finally, the applicability of the STMCM to a nonlinear mathematical model of tumour growth is demonstrated by studying the accuracy and convergence properties of the method.

The summary of the thesis and an outlook are given in Chapter 8 on page 149.

"Make everything as simple as possible, but not simpler."
-- Albert Einstein

"The simplest things are often the truest."
-- Richard Bach

CHAPTER 2

Basic concepts and Methodology

Originally the Moving Least Squares (MLS) method and its Interpolating counterpart (IMLS) were introduced by Lancaster andŠ alkauskas in their paper [109] and later on it was extended and became a book [110], wherein it is used for interpolation and smoothing of scattered data. The former technique became the basis for many of the meshfree methods developed in recent years. The latter has not been widely used mainly because of the singularity-by-construction problem, which, however, can be solved by means of a regularization [150, 108].

In the following we deduce the MLS and IMLS methods via minimization of the weighted least squares functional and afterwards discuss the regularization technique for IMLS method in detail in Section 2.6.2 on page 21.

2.1 Deduction of the method

For given data of a function $u : \Omega \subset \mathbb{R}^d \mapsto \mathbb{R}$ and a fixed point $\mathbf{x}^* \in \Omega$ one seeks a local approximation of u around $\mathbf{x}^*$ in form of the following function

$$u^{\text{loc}}(\mathbf{x}, \mathbf{x}^*) = \mathscr{L}_{\mathbf{x}^*} u(\mathbf{x}) := \mathbf{b}^{\mathsf{T}}(\mathbf{x}) \cdot \mathbf{a}(\mathbf{x}^*) = \sum_{i=1}^{m} a_i(\mathbf{x}^*) b_i(\mathbf{x}), \tag{2.1}$$

which is the best approximation of $u(\mathbf{x})$ in a certain least squares sense having the property

$$u^{\text{loc}}(\mathbf{x}, \mathbf{x}^*) = \begin{cases} u(\mathbf{x}), & \forall \mathbf{x} \in B_\rho(\mathbf{x}^*) \\ 0, & \text{else} \end{cases}, \tag{2.2}$$

where the open ball $B_\rho(\mathbf{x}^*)$ with the radius ρ is defined as

$$B_\rho(\mathbf{x}^*) := \{\mathbf{x} : ||\mathbf{x} - \mathbf{x}^*|| < \rho, \mathbf{x} \in \Omega\}. \tag{2.3}$$

The vector $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^m$ is the complete basis (its choice is outlined in Section 2.4 on page 16) and $\mathbf{a}(\mathbf{x}^*) \in \mathbb{R}^m$ is a vector of unknown coefficients.

Then, the so-called "moving procedure" is introduced in order to achieve a global approximation. Since the point $\mathbf{x}^*$ can be chosen arbitrarily, it is *moved* over the whole domain Ω and hence a global approximation function defined as

$$u^{\text{appx}}(\mathbf{x}) = \mathscr{G}u(\mathbf{x}^*) := \lim_{\mathbf{x}^* \to \mathbf{x}} \mathscr{L}_{\mathbf{x}^*} u(\mathbf{x})$$

is obtained with the help of (2.1) as follows

$$\mathscr{G}u(\mathbf{x}^*) = \mathscr{L}_{\mathbf{x}^*} u(\mathbf{x}^*) = \mathbf{b}^\mathsf{T}(\mathbf{x}^*) \cdot \mathbf{a}(\mathbf{x}^*) = \sum_{i=1}^{m} a_i(\mathbf{x}^*) b_i(\mathbf{x}^*). \tag{2.4}$$

The operator $\mathscr{G}$ in (2.4) can be interpreted as the globalization of the local approximation operator $\mathscr{L}_{\mathbf{x}^*}$ from (2.1) through the moving procedure [109].

2.2 (Interpolating) Moving Least Squares

Let $\mathbf{x}_i = (x_{1i}, x_{2i}, \ldots, x_{di}, t_i)^\mathsf{T} \in \mathbb{R}^{d+1}$, $i = 1, \ldots, N$ be a set of N data points in a space-time domain (d dimensions for space and one for time). Then, given N pairs $(\mathbf{x}_i, u_i), i = 1, \ldots, N$, the Moving Least Squares method (MLS) is defined by assigning weights $w_i(\mathbf{x}) := w(||\mathbf{x} - \mathbf{x}_i||), i = 1, \ldots, N$ to each point $\mathbf{x}$ and then asking for the minimum of the functional

$$E_\mathbf{x}(\mathbf{a}) = \sum_{i=1}^{N} w(||\mathbf{x} - \mathbf{x}_i||)(p(\mathbf{x}_i) - u_i)^2, \tag{2.5}$$

where $p(\mathbf{x}) := \mathbf{b}^\mathsf{T} \cdot \mathbf{a}$ is a polynomial with basis $\mathbf{b}(\mathbf{x}) := (b_1(\mathbf{x}), \ldots, b_m(\mathbf{x}))^\mathsf{T}$ and coefficients $\mathbf{a} := (a_1, \ldots, a_m)^\mathsf{T} \in \mathbb{R}^m$.

Many different basis functions are reported in literature [68], and their choice depends, inter alia, on the problem under consideration. One of the very frequent choices is the use of monomials of a certain order due to its differentiability. In the space-time approach presented in Section 3 on page 35 we are free to choose different (or the same) order of space and time discretizations. The choice of the basis $\mathbf{b}$ is discussed in Section 2.4 on page 16.

For a specific case when the weights $w_i(\mathbf{x}) \equiv 1 \; \forall\, i$, one obtains from (2.5) the well-known Least Squares approximation. So why did the present technique get its name? What is *moving* here? We will answer these questions momentarily after looking at the solution of the minimization problem (2.5).

The choice of the weight function w in (2.5) is crucial for the art of the method we end up with. We will discuss its choice in detail in Section 2.5. The functional $E_x(\mathbf{a})$ from (2.5) is minimized by setting its derivative with respect to the coefficients $\mathbf{a}$ equal to zero, i.e. $\frac{\partial E_x(\mathbf{a})}{\partial \mathbf{a}} = 0$ and we get the following system of equations

$$
\begin{aligned}
\frac{\partial E_x(\mathbf{a})}{\partial a_1} &= 0 \quad \Rightarrow \quad \sum_{i=1}^{N} w_i(\mathbf{x}) \cdot 2b_1(\mathbf{x}_i) \cdot \left[\, \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} - u_i \,\right] = 0, \\
\frac{\partial E_x(\mathbf{a})}{\partial a_2} &= 0 \quad \Rightarrow \quad \sum_{i=1}^{N} w_i(\mathbf{x}) \cdot 2b_2(\mathbf{x}_i) \cdot \left[\, \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} - u_i \,\right] = 0, \\
&\ \vdots \qquad\quad \vdots \qquad\quad \vdots \\
\frac{\partial E_x(\mathbf{a})}{\partial a_m} &= 0 \quad \Rightarrow \quad \sum_{i=1}^{N} w_i(\mathbf{x}) \cdot 2b_m(\mathbf{x}_i) \cdot \left[\, \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} - u_i \,\right] = 0,
\end{aligned}
$$

which in vector notation writes as

$$
\begin{aligned}
\sum_{i=1}^{N} w_i(\mathbf{x}) \cdot 2\mathbf{b}(\mathbf{x}_i) \cdot \left[\, \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} - u_i \,\right] &= 0 \quad \Leftrightarrow \\
2\sum_{i=1}^{N} w_i(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}_i) \cdot \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} - 2\sum_{i=1}^{N} w_i(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}_i) \cdot u_i &= 0 \quad \Leftrightarrow \\
\sum_{i=1}^{N} w_i(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}_i) \cdot \mathbf{b}^{\mathsf{T}}(\mathbf{x}_i) \cdot \mathbf{a} = \sum_{i=1}^{N} w_i(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}_i) \cdot u_i. & \qquad (2.6)
\end{aligned}
$$

Rewriting the equation (2.6) in a matrix form, the minimum of the functional (2.5) is given as a solution of the normal equations system

$$
\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B} \cdot \mathbf{a} = \mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{u}, \qquad (2.7)
$$

where the basis matrix is defined as

$$
\mathbf{B} := \begin{pmatrix} b_1(\mathbf{x}_1) & b_2(\mathbf{x}_1) & \cdots & b_m(\mathbf{x}_1) \\ b_1(\mathbf{x}_2) & b_2(\mathbf{x}_2) & \cdots & b_m(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ b_1(\mathbf{x}_N) & b_2(\mathbf{x}_N) & \cdots & b_m(\mathbf{x}_N) \end{pmatrix} \in \mathbb{R}^{N\times m}
$$

and the weight matrix as

$$
\begin{aligned}
\mathbf{W}(\mathbf{x}) &:= \operatorname{diag}\left(w(\mathbf{x}-\mathbf{x}_1), w(\mathbf{x}-\mathbf{x}_2), \dots, w(\mathbf{x}-\mathbf{x}_N)\right) \\
&=: \operatorname{diag}\left(w_1(\mathbf{x}), w_2(\mathbf{x}), \dots, w_N(\mathbf{x})\right) \in \mathbb{R}^{N\times N}.
\end{aligned}
$$

Note that $\mathbf{W}$ depends on the point $\mathbf{x}$. Hence, the vector of coefficients $\mathbf{a}$ will, in general, depend on $\mathbf{x}$ too.

Since $\mathbf{a}(\mathbf{x})$ is the solution of (2.7) we can write

$$
\mathbf{a}(\mathbf{x}) = \left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{u}, \qquad (2.8)
$$

and the interpolation function can be written in the form

$$\begin{aligned} p(\mathbf{x}) &= \mathbf{b}^{\mathsf{T}}(\mathbf{x}) \cdot \mathbf{a}(\mathbf{x}) \\ &=< \mathbf{b}(\mathbf{x}), \left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{u} > \\ &=< \left[\left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\right]^{\mathsf{T}} \mathbf{b}(\mathbf{x}), \mathbf{u} > \\ &=< \mathbf{W}(\mathbf{x})\mathbf{B} \left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{b}(\mathbf{x}), \mathbf{u} > . \end{aligned} \tag{2.9}$$

One notices that now the coefficients **a** depend on **x**, which is not the case for a conventional Least Squares approximation! This dependence appears due to the definition of the weights $w_i(\mathbf{x})$. Hence, the coefficients of the approximating (or interpolating) polynomial $p(\mathbf{x})$ are *different* for every point **x**, meaning that as we *move* throughout the domain, we obtain a *different* polynomial.

Defining the *kernel functions* (also called 'shape functions' in literature)

$$\boldsymbol{\varphi}(\mathbf{x}) := \mathbf{W}(\mathbf{x})\mathbf{B} \left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{b}(\mathbf{x}), \tag{2.10}$$

the sought approximation function can be written as

$$u^{\text{appx}}(\mathbf{x}) = p(\mathbf{x}) =< \boldsymbol{\varphi}(\mathbf{x}), \mathbf{u} >= \boldsymbol{\varphi}^{\mathsf{T}}(\mathbf{x}) \cdot \mathbf{u} = \sum_{i=1}^{N} u_i \varphi_i(\mathbf{x}). \tag{2.11}$$

Looking at the dimensions of the matrices involved in the kernel function (2.10)

$$\underbrace{\boldsymbol{\varphi}(\mathbf{x})}_{N\times 1} = \underbrace{\mathbf{W}(\mathbf{x}) \cdot \mathbf{B}}_{(N\times m)} \cdot \underbrace{\left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1}}_{(m\times m)} \cdot \underbrace{\mathbf{b}(\mathbf{x})}_{(m\times 1)},$$

it becomes apparent that the matrix $\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}$ of the size $m \times m$ is to be inverted at each point **x** where the values of the kernel function $\boldsymbol{\varphi}(\mathbf{x})$ are needed. For the basis of order q we note that $m = q + 1$ in 1d and $m = \frac{1}{2}(q+1)(q+2)$ in 2d.

A special case of the kernel functions is the *Shepard interpolant*, which is obtained for the case $m = 1$, i.e. for a constant basis $\mathbf{b}(\mathbf{x}) \equiv 1$. In this case, the matrix to be inverted in (2.10) becomes a diagonal one, reducing herewith the computational cost. However, the higher order derivatives vanish at the nodes [184], making it useless for the collocation solution of PDEs that will be considered in further sections. This property of the Shepard interpolant is often called a 'flat spot phenomenon', see e.g. [184, 110].

Obviously, if a higher concistency order is desired, i.e. for higher values of m, the computational time rises due to the large number of evaluation points. This and some other important aspects concerning the inversion of the matrix are discussed in detail in Section 2.6 on page 19 and important properties of the kernel functions are presented in the Section 2.8 on page 29.

2.3 Comparison with other Least Squares methods

The basic idea behind every least square method is to minimize the sum of the squared error weighted with some function w, i.e. the distances between the real function values $u(\mathbf{x})$ and the approximated ones $u^{\text{appx}}(\mathbf{x})$, with respect to free parameters:

$$E := \sum_{i=1}^{N} w \, [\; u(\mathbf{x}) - u^{\text{appx}}(\mathbf{x}) \;]^2 \rightarrow \min .$$

The approximation $u^{\text{appx}}(\mathbf{x})$ is sought in form of a polynomial $u^{\text{appx}}(\mathbf{x}) := p(\mathbf{x}) = \mathbf{b}^{\mathsf{T}}(\mathbf{x}) \cdot \mathbf{a}$, where $\mathbf{b}$ is some basis and $\mathbf{a}$ are the free parameters.

This minimization task is well-posed, see e.g. [62], i.e. there exists one and only one polynomial for which the minimum of the functional E exists, if the order of the polynomial is less or equal to the number of points N.

Depending on the choice of this weight function w one can classify the least squares methods as follows:

- $w \equiv 1$: *Least Squares (LS)*
  One ends up with the standard least squares method. The matrix $\mathbf{W}(\mathbf{x})$ from (2.7) becomes the identity matrix and the solution to the minimization problem (2.7) itself becomes the system of normal equations

  $$\mathbf{B}^{\mathsf{T}}\mathbf{B} \cdot \mathbf{a} = \mathbf{B}^{\mathsf{T}}\mathbf{u}.$$

  Certain values of the unknown free parameters $\mathbf{a}$ of the approximation polynomial are computed "once and forever", which implies the main disadvantage of this method: if the number of evaluation points is much greater than the order of the polynomial (which is mostly the case in practice), i.e. $N \gg m$, then the approximation is getting worse very rapidly.

- $w = w_j(\mathbf{x}_i)$: *Weighted Least Squares (WLS)*
  In this case for each evaluation point $\mathbf{x}$ a fixed weight function w_j is chosen. It has a maximum at the point $\mathbf{x}_j \neq \mathbf{x}$. The choice of the appropriate weight function is rather difficult and problem-dependent. The system of normal equations is in this case

  $$\mathbf{B}^{\mathsf{T}}\mathbf{W}\mathbf{B} \cdot \mathbf{a} = \mathbf{B}^{\mathsf{T}}\mathbf{W}\mathbf{u}.$$

  In order to assure that the jumps in the approximation do not occur, one must choose only one w_j for each point $\mathbf{x}$. But for a different $\mathbf{x}$ different w_j has to be used.

- $w = w(\mathbf{x} - \mathbf{x}_i)$: *Moving Least Squares (MLS)*
  In the MLS case, the weight function w has a certain shape and support size, and it is moved over the whole domain. It has its maximum at an arbitrary point $\mathbf{x}$ where

the function values have to be computed. We repeat the equation (2.7) at this place for completeness

$$\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\cdot\mathbf{a} = \mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{u}.$$

Since the matrix $\mathbf{W}(\mathbf{x})$ depends on $\mathbf{x}$, so does the vector of coefficients $\mathbf{a}$. Hence we write $\mathbf{a}(\mathbf{x})$ instead of $\mathbf{a}$. The inversion of the matrix in (2.10) is required at every evaluation point $\mathbf{x}$. Hence in general, one does not get one interpolation polynomial, but rather a different polynomial with different coefficients $\mathbf{a}(\mathbf{x})$ at every point $\mathbf{x} \in \Omega$.

2.4 Basis functions

The choice of basis functions is very important for accurate discretization by numerical methods. We would like our basis functions to enjoy the following properties: ease of computation, completeness, and convergence. By completeness we mean that any given order of accuracy can be achieved by taking the order of the basis function large enough.

The simplest and widely used basis functions are the polynomial basis functions of degree m:

$$\mathbf{b}(x) := (1, x, x^2, \ldots, x^m)^{\mathsf{T}} \qquad \text{in 1d} \tag{2.12}$$

and

$$\mathbf{b}(x,y) := (1, x, y, x^2, xy, y^2, \ldots, y^m)^{\mathsf{T}} \qquad \text{in 2d,} \tag{2.13}$$

and similarly for higher dimensions.

Two types of basis polynomial spaces, both including the complete basis of the subspace of polynomials of degree m, are usually used in the literature on meshfree methods:

(i) the set of polynomials P_m of total degree $q \leq m$,

(i) the set of polynomials P'_m of degree $q' \leq m$ for each variable.

This choice is characteristic for an a priori convergence rate of the method [93].

According to [25], for 95% of all applications the best choice of basis functions are ordinary Fourier series. The most important special case of those are the Chebyshev (also spelled sometimes as Chebyshov, Tschebyscheff, or Čebyshev) polynomials defined by

$$T_n(\cos\theta) = \cos(n\theta). \tag{2.14}$$

Zeros occur for $x = \cos\left(\frac{\pi\left(k-\frac{1}{2}\right)}{n}\right)$, $k = 1, 2, \ldots, n$ and extrema occur for $x = \cos\left(\frac{\pi l}{n}\right)$, $l = 0, 1, \ldots, n$.

The Chebyshev polynomials are extremely robust and give good results in many situations. However, they require a special arrangement of the nodes.

- The choice of Chebyshev or other orthogonal polynomials, e.g. Jacobi, Legendre, Hermite polynomials, etc., which are commonly used in mesh-based methods for numerical integration by quadratures, is possible.

  However, in the approach presented in this thesis there is no need for numerical integration since, as we shall see in further sections, the equations are discretized in strong form. Therefore, in our numerical computations we will use polynomials of certain order of the form (2.12)–(2.13) as basis functions.

- If an a priori knowledge on solution characteristics is available, one can use this information and incorporate the appropriate functions into the basis, e.g. one may include sinusoidal and exponential functions when solving ordinary differential equations with periodic solutions.

2.5 Weight functions

Different weights are reported in the literature (e.g. splines, Gaussian curves, etc.) and in case of approximation problems their specific expressions differ by grades of locality and number of parameters but are only of slight importance for the quality of approximation. There is, however, an essential difference between interpolatory and non-interpolatory weight functions. The choice between these two cases also defines which method we end up with – Moving Least Squares (MLS) or Interpolating Moving Least Squares (IMLS).

Moving Least Squares (MLS) functions are sometimes erroneously referred to in the literature on meshfree methods as *interpolation*, whereas these are only *approximative*, since they do not pass through the data. In fact, the choice of the weights is decisive. In order to get the interpolation the condition

$$\lim_{\mathbf{x}\to\mathbf{x}_i} w(\|\mathbf{x}-\mathbf{x}_i\|)=\infty \tag{2.15}$$

must be satisfied [109]. This requirement leads to the problem of singularity, since the weights are simply not defined at the collocation point $\mathbf{x}=\mathbf{x}_i$, where the kernel functions defined in (2.10) must be evaluated. We present the technique to overcome this problem in Section 2.6.2 on page 21.

Firstly, let us look at the approximating case which is the starting point of most of the meshfree methods.

2.5.1 Approximation case :: MLS

In the approximation, i.e. non-interpolatory case popular weights are continuously differentiable spline-functions of the 3rd and 4th order, respectively:

$$w_3(\zeta) \;:=\; \begin{cases} 1-3\zeta^2+2\zeta^3 & , \quad \zeta\leqslant 1 \\ 0 & , \quad \zeta>1 \end{cases}, \tag{2.16}$$

$$w_4(\zeta) \;:=\; \begin{cases} 1-6\zeta^2+8\zeta^3-3\zeta^4 & , \quad \zeta\leqslant 1 \\ 0 & , \quad \zeta>1 \end{cases}, \tag{2.17}$$

where $\zeta := \frac{\|\mathbf{x}-\mathbf{x}_i\|}{\rho}$ and ρ is the so-called dilatation parameter. The latter defines the support of the weight function and is crucial for the accuracy of the method. The Figure 2.1(a) represents a MLS weight function using fourth order spline.

In [110] exponential functions as weight functions are suggested. These do not have, however, a compact support. Note that (2.16) and (2.17) do have a compact support and that they have just one free parameter ρ.

Note further that the matrix $\mathbf{B}^{\mathsf{T}}\mathbf{W}(x)\mathbf{B}$ in (2.7) will be singular if the support of the weight functions is too small. The 'rule of thumb' found in the literature is: the supports must be as small as possible, but they must overlap so that the whole computational domain of interest be covered, see e.g. [68]. Guidelines on choosing the size of the support of the weight functions as well as other parameters of the proposed approach are given in Section 3.5 on page 43.

2.5.2 Interpolation case :: IMLS

If exact interpolation at the nodes x_i is required, then according to [109] the weight functions must satisfy the interpolation property (2.15). Popular weight functions are therefore $w(\zeta) := \zeta^{-\alpha}$ or $w(\zeta) := \zeta^{-\alpha} \cdot \mathrm{e}^{-\zeta^{\alpha}}$, where $\alpha \geq 2$, even. The requirement on α ensures that the far away points do not obtain too much weight.

In this case, however, the weight functions do not have a local support, which means that the sums involved in the construction fo the kernel function $\boldsymbol{\varphi}$ will go over all computational points in the domain, which is disadvantageous as the number of points rises, since the resulting matrices will not be sparse. The supports can, of course, be cut off so that the support is local.

Another possibility is to take weight functions from the non-interpolatory case (2.16) or (2.17) and add a singularity, e.g. $w(\zeta) := \frac{w_3(\zeta)}{\zeta^{\alpha}}$ or $w(\zeta) := \frac{w_4(\zeta)}{\zeta^{\alpha}}$. Figure 2.1(b) shows the IMLS weight function for $\alpha = 6$.

We choose for our computations the locally supported *singular* weights

$$w_i(\mathbf{x}) := \begin{cases} \frac{1}{\|\mathbf{x}-\mathbf{x}_i\|^{\alpha}}, & \alpha = 2k,\, k \in \mathbb{N} \quad , \|\mathbf{x}-\mathbf{x}_i\| < \rho \\ 0 & \quad , \text{else} \end{cases}, \tag{2.18}$$

with the dilatation parameter ρ which, in general, can vary throughout the domain, so that our method belongs to the class of Interpolating Moving Least Squares (IMLS) methods.

In Section 2.6.2 we will show that the IMLS kernel functions, i.e. the kernel functions constructed from the interpolatory method, fulfill the Kronecker delta property, i.e.

$$\varphi_j(x_i) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} .$$

Very few publications applying the IMLS method are up to now available. Maisuradze et al. apply the IMLS method in [138] for fitting surfaces of the potential energy in chemical

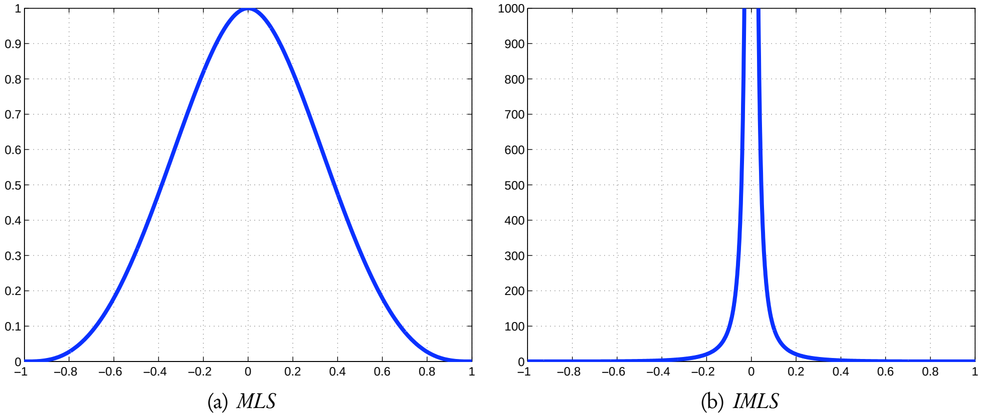


Figure 2.1 | Weight functions in MLS and IMLS case.

application. In [108] Kunle compares the accuracy of the two approaches described above and shows that the approximating MLS method is less suitable for approximating higher order derivatives with steep gradients than its interpolating counterpart. Therefore, in the following we only consider the IMLS-method and investigate its accuracy and convergence properties when applied amongst others to boundary layer and shock propagation problems.

2.6 Singularity-by-Construction

In the non-interpolatory case the inversion of the matrix in (2.10) does not cause many problems. Since this is a small $(m \times m)$-matrix, where m depends on the order of the basis, the inversion can be done via the LU-decomposition with pivoting with relatively little effort, see e.g. [26, 68]. Nonetheless, if the matrix is ill-conditioned , the LU-decomposition may lead to false results [108]. One should therefore use QR- or Singular Value Decomposition to get admissible results [73].

We run into much more trouble if we want to use Interpolating Moving Least Squares. In this case the weight function matrix $\mathbf{W}(\mathbf{x})$ contains a singularity in the ℓ-th row: $w_\ell(\mathbf{x}_\ell)$ is simply not defined. Due to this fact the matrix $\mathbf{B}^\top \mathbf{W}(\mathbf{x})\mathbf{B}$ in (2.10) is also singular. In order to compute the inverse of this singular matrix we follow the idea introduced in [108].

The problems of finding (pseudo)-inverses of ill-conditioned matrices were an area of research of mathematicians back in the 70s already [24], and a number of specific algorithms for certain types of matrices have been developed, e.g. [173, 83, 82, 84, 86]. Regularization tools for analysis and solution of discrete ill-posed problems are also available [81].

Since finding an inverse of an ill-conditioned matrix is essential for our method, let us look at a powerful technique for dealing with matrices that are numerically very close to singular – the Singular Value Decomposition – our method of choice.

2.6.1 Singular Value Decomposition

The Singular Value Decomposition (SVD) is a set of techniques to deal with numerically close-to-singular and singular matrices. In the cases where the direct Gaussian elimination algorithm and the LU-Decomposition do not succeed to provide an admissible result, the SVD will not only detect the problem, but will also solve it giving a useful numerical answer [169].

The method is based on the following lemma of linear algebra.

Lemma 2.1 *Any matrix* $\mathbf{A}$ *of size* $(m \times n)$, $m \geq n$ *can be decomposed into a product*

$$\mathbf{A} = \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^T, \tag{2.19}$$

where $\mathbf{U}$ *is a column-orthogonal matrix of size* $(m \times n)$, *the matrix* Σ *is a diagonal* $(n \times n)$*-matrix with positive of zero elements (*singular values*), and the matrix* $\mathbf{V}$ *is an orthogonal matrix of size* $(n \times n)$.

Proof The proof is beyond the focus of this thesis, it can be found in [73]. ☐

The matrices $\mathbf{U}$ and $\mathbf{V}$ are each orthogonal, i.e. their columns are orthonormal, and $\mathbf{V}$ is also row-orthogonal since it is square. Hence,

$$\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{V}^\mathsf{T}\mathbf{V} = \mathbf{V}\mathbf{V}^\mathsf{T} = \mathbf{I},$$

where $\mathbf{I}$ is an identity matrix of size $(n \times n)$.

According to [169] the SVD algorithm is very stable, and it is very unusual for it ever to misbehave. The decomposition (2.19) can be performed, independent of how singular the matrix is, and is unique up to:

- making the same permutations of the columns of $\mathbf{U}$, elements of Σ, and columns of $\mathbf{V}$, or
- forming linear combinations of any columns of $\mathbf{U}$ and $\mathbf{V}$ whose corresponding elements of Σ are exactly equal.

In our derivations we shall only concentrate on finding the inverse of a square matrix $\mathbf{A}_{(n\times n)}$ via SVD, as only this is needed in our IMLS-based method. The composition of rectangular matrices is described in [169].

If the matrix is square of size n, then the product matrices $\mathbf{U}, \mathbf{V}$ and Σ are also square of the same size. Moreover, their inverses are trivial to compute: the matrix Σ is diagonal, so its inverse is a diagonal matrix with reciprocal values of Σ_i, $i = 1, \ldots, n$, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, so $\mathbf{U}^{-1} = \mathbf{U}^\mathsf{T}$ and $\mathbf{V}^{-1} = \mathbf{V}^\mathsf{T}$.

Hence, from (2.19) we immediately obtain

$$\mathbf{A}^{-1} = \mathbf{V} \cdot \mathsf{diag}(1/\Sigma_i) \cdot \mathbf{U}^\mathsf{T}. \tag{2.20}$$

The only possible problem with this approach is if one or more of the Σ_i values are zero (or numerically so small that its value is undetermined due to a restricted machine accuracy). The rate of singularity of a matrix is best reflected by its *condition number* which is formally defined as the ratio of the largest and the smalles values of Σ_i, $i = 1, \ldots, n$.

The matrix **A** is *singular* if its condition number

$$\mathsf{cond}(\mathbf{A}) = \infty,$$

and is *ill-conditioned* if its reciprocal approaches the machine accuracy ϵ, e.g. 10^{-6} for single and 10^{-12} for double precision

$$\frac{1}{\mathsf{cond}(\mathbf{A})} \to \epsilon.$$

An important step in the SVD procedure, therefore, is to replace $1/\Sigma_i$ by zero if $\Sigma_i = 0$. In practice we zero those singular values that are smaller than the machine accuracy of the used computer $2.2 \cdot 10^{-16}$.

▶ We will not go into detail at this stage and the interested reader is referred to [73, 169] for details. In should only be pointed out that although it seems quite wrong to zero one of the singular values, since it refers to one of the linear combinations of the equations to be solved, it turns out to be legitimate. With this procedure the combination of equations, which can (at best) be useless, is being neglected.

2.6.2 Regularization technique

The challenge with interpolatory weights presented in Section 2.5 on page 17 is obviously the singularity appearing in the matrix **W**, namely the weight $w_\ell(\mathbf{x}_\ell)$, which can only be dealt with by means of regularization.

The author in [108] outlines the approach we follow here already. However, in his work Kunle avoids notations from linear algebra and uses a fairly inelegant sum notation which makes conclusions concerning higher order derivatives of the kernel functions impossible and which does not provide any structural understanding.

The description of attacking the singularity problem is presented and published by the author in [150, 152, 151], and in the following we take these representations with some extensions as a pattern and not only clearly reveal the structure of the kernel functions, but also provide the structure of derivatives of arbitrary order.

In order to cure the singularity problem we introduce a regularization due to a small positive number ε in the form of the regularized weight function matrix $\mathbf{W}(\mathbf{x}+\varepsilon)$. Then, the equation (2.10) reads as

$$\boldsymbol{\varphi}(\mathbf{x}+\varepsilon) = \mathbf{W}(\mathbf{x}+\varepsilon)\mathbf{B}\left(\mathbf{B}^\mathsf{T}\mathbf{W}(\mathbf{x}+\varepsilon)\mathbf{B}\right)^{-1}\mathbf{b}(\mathbf{x}+\varepsilon). \tag{2.21}$$

We proceed with the following lemma and theorem.

Lemma 2.2 *An inverse of a matrix perturbed with a change, which can be written in a form* $\mathbf{u}\otimes\mathbf{v}$*, can be computed by the Sherman-Morrison formula given by*

$$(\mathbf{A}+\mathbf{u}\otimes\mathbf{v})^{-1}=\mathbf{A}^{-1}-\frac{(\mathbf{A}^{-1}\cdot\mathbf{u})\otimes(\mathbf{v}\cdot\mathbf{A}^{-1})}{1+\mathbf{v}\cdot\mathbf{A}^{-1}\cdot\mathbf{u}}.$$

Proof See [73] for a detailed proof. □

- ▶ The Sherman-Morrison formula is a simplified version of a Woodbury formula wherein the change can be written as a product of any (dimensionally compatible) matrices $\mathbf{U}$ and $\mathbf{V}$, rather than the product of rows $\mathbf{u}$ and $\mathbf{v}$, i.e.

$$(\mathbf{A}+\mathbf{U}\mathbf{V}^{\mathsf{T}})^{-1}=\mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{U}\left(\mathbf{I}+\mathbf{V}^{\mathsf{T}}\mathbf{A}^{-1}\mathbf{U}\right)^{-1}\mathbf{V}^{\mathsf{T}}\mathbf{A}^{-1},$$

where $\mathbf{I}$ is an identity matrix of appropriate size.

Theorem 2.3 *The matrix* $\mathbf{B}^T\mathbf{W}(\mathbf{x})\mathbf{B}$ *is regular and its inverse at the point* $\mathbf{x}_\ell$ *is given by*

$$\left(\mathbf{B}^T\mathbf{W}(\mathbf{x}_\ell)\mathbf{B}\right)^{-1}=\mathbf{R}^{-1}-\frac{1}{\mathbf{b}^T(\mathbf{x}_\ell)\,\mathbf{R}\,\mathbf{b}(\mathbf{x}_\ell)}\left(\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)\cdot\left(\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)^T,$$

with a regular and symmetric matrix

$$\mathbf{R}:=\begin{pmatrix}\sum_{\substack{i=1\\i\neq\ell}}^N b_1(\mathbf{x}_i)w_i(\mathbf{x}_\ell)b_1(\mathbf{x}_i) & \cdots & \sum_{\substack{i=1\\i\neq\ell}}^N b_1(\mathbf{x}_i)w_i(\mathbf{x}_\ell)b_m(\mathbf{x}_i)\\ \vdots & \ddots & \vdots\\ \sum_{\substack{i=1\\i\neq\ell}}^N b_m(\mathbf{x}_i)w_i(\mathbf{x}_\ell)b_1(\mathbf{x}_i) & \cdots & \sum_{\substack{i=1\\i\neq\ell}}^N b_m(\mathbf{x}_i)w_i(\mathbf{x}_\ell)b_m(\mathbf{x}_i)\end{pmatrix}.$$

Proof Regularizing the weight matrix with ε and splitting it up into two parts we get

$$\begin{aligned}\mathbf{W}(\mathbf{x}_\ell+\varepsilon) &= \mathrm{diag}(w_1(\mathbf{x}_\ell+\varepsilon)\dots,w_{\ell-1}(\mathbf{x}_\ell+\varepsilon),\underbrace{\quad 0\quad}_{\ell^{\text{th}}\text{ term omitted}},w_{\ell+1}(\mathbf{x}_\ell+\varepsilon),\dots,w_N(\mathbf{x}_\ell+\varepsilon))\\ &+ \mathrm{diag}(0,\dots,0,\underbrace{w_\ell(\mathbf{x}_\ell+\varepsilon)}_{\ell^{\text{th}}\text{ position}},0,\dots,0).\end{aligned}$$

Hence,

$$\begin{aligned}&\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x}_\ell+\varepsilon)\mathbf{B}=\\ &\underbrace{\begin{pmatrix}\sum_{\substack{i=1\\i\neq\ell}}^N b_1(\mathbf{x}_i)w_i(\mathbf{x}_\ell+\varepsilon)b_1(\mathbf{x}_i) & \cdots & \sum_{\substack{i=1\\i\neq\ell}}^N b_1(\mathbf{x}_i)w_i(\mathbf{x}_\ell+\varepsilon)b_m(\mathbf{x}_i)\\ \vdots & \ddots & \vdots\\ \sum_{\substack{i=1\\i\neq\ell}}^N b_m(\mathbf{x}_i)w_i(\mathbf{x}_\ell+\varepsilon)b_1(\mathbf{x}_i) & \cdots & \sum_{\substack{i=1\\i\neq\ell}}^N b_m(\mathbf{x}_i)w_i(\mathbf{x}_\ell+\varepsilon)b_m(\mathbf{x}_i)\end{pmatrix}}_{=:\mathbf{R}_\varepsilon}\\ +\;& w_\ell(\mathbf{x}_\ell+\varepsilon)\cdot\begin{pmatrix}b_1(\mathbf{x}_\ell)b_1(\mathbf{x}_\ell) & \cdots & b_1(\mathbf{x}_\ell)b_m(\mathbf{x}_\ell)\\ \vdots & \ddots & \vdots\\ b_m(\mathbf{x}_\ell)b_1(\mathbf{x}_\ell) & \cdots & b_m(\mathbf{x}_\ell)b_m(\mathbf{x}_\ell)\end{pmatrix}.\end{aligned} \tag{2.22}$$

Using the Sherman-Morrison formula from Lemma 2.2 on the facing page it follows

$$\begin{aligned}
\left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x}_\ell)\mathbf{B}\right)^{-1} &= \mathbf{R}_\varepsilon^{-1} - \frac{w_\ell(\mathbf{x}_\ell+\varepsilon)}{1+w_\ell(\mathbf{x}_\ell+\varepsilon)\mathbf{b}^{\mathsf{T}}(\mathbf{x}_\ell)\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)}\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)^{\mathsf{T}} \\
&= \mathbf{R}_\varepsilon^{-1} - \frac{1}{\frac{1}{w_\ell(\mathbf{x}_\ell+\varepsilon)}+\mathbf{b}^{\mathsf{T}}(\mathbf{x}_\ell)\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)}\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)^{\mathsf{T}} \\
&= \mathbf{R}_\varepsilon^{-1} - \frac{1}{\frac{1}{w_\ell(\mathbf{x}_\ell+\varepsilon)}+\sigma}\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)\left(\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)^{\mathsf{T}}, \qquad (2.23)
\end{aligned}$$

where

$$\sigma := \mathbf{b}^{\mathsf{T}}(\mathbf{x}_\ell)\mathbf{R}_\varepsilon^{-1}\mathbf{b}(\mathbf{x}_\ell)\,. \qquad (2.24)$$

Since it holds

$$\lim_{\varepsilon\to 0} w_\ell(\mathbf{x}_\ell+\varepsilon) = w_\ell(\mathbf{x}_\ell) = \infty \quad \Longrightarrow \quad \lim_{\varepsilon\to 0}\frac{1}{w_\ell(\mathbf{x}_\ell+\varepsilon)} = 0\,,$$

we obtain from (2.22)

$$\lim_{\varepsilon\to 0}\mathbf{R}_\varepsilon = \mathbf{R}\,,$$

and herewith the desired result. □

2.6.3 Derivatives and nodal values of the kernel functions

Now we are ready to derive the formulae for the derivatives of the IMLS kernel function (2.10) needed for solving differential equations by the approach presented in later chapters and to compute the values at the nodes.

In order to do so we shall consider two different cases, namely

$$\varphi_\ell(\mathbf{x}_\ell+\varepsilon)$$

and

$$\varphi_k(\mathbf{x}_\ell+\varepsilon), \quad k\neq\ell.$$

Theorem 2.4 *Let the derivative operator w.r.t. some arbitrary variable* $\mathbf{z}$ *(it can be space or time) in a multi-index notation be defined as*

$$\mathscr{D}_{\mathbf{z}}^{\alpha} := \frac{\partial^{|\alpha|}}{\partial z_1^{\alpha_1}\cdot\partial z_2^{\alpha_2}\cdot\ldots\cdot\partial z_n^{\alpha_n}}, \qquad |\alpha| = \alpha_1+\alpha_2+\ldots+\alpha_n.$$

Then, for each node $\mathbf{x}_\ell$, $\ell = 1, 2, \dots, N$ *for kernel functions and their derivatives it holds*

$$\begin{aligned}
\varphi_\ell(\mathbf{x}_\ell) &= 1, \\
\mathscr{D}^\alpha_{\mathbf{z}} \varphi_\ell(\mathbf{x}_\ell) &= \frac{1}{\mathbf{b}^T(\mathbf{x}_\ell)\,\mathbf{R}\,\mathbf{b}(\mathbf{x}_\ell)} \mathbf{b}^T(\mathbf{x}_\ell)\mathbf{R}^{-1} \cdot \mathscr{D}^\alpha_{\mathbf{z}} \mathbf{b}(\mathbf{x}_\ell).
\end{aligned}$$

Proof Without loss of generality we consider an one-dimensional case, i.e. $\mathbf{x}_\ell \equiv x_\ell$.

Considering the regularized version of the kernel function (2.10) evaluated at the node x_ℓ, and inserting the inverse of the matrix (2.23) from Theorem 2.3 we obtain

$$\begin{aligned}
\varphi_\ell(x_\ell+\varepsilon) &= \mathbf{W}(x_\ell+\varepsilon)\mathbf{B}\left(\mathbf{B}^\mathsf{T}\mathbf{W}(x_\ell+\varepsilon)\mathbf{B}\right)^{-1}\mathbf{b}(x_\ell+\varepsilon) \qquad (2.25)\\
&= \mathbf{W}(x_\ell+\varepsilon)\mathbf{B}\left(\mathbf{R}^{-1} - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma}\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)\cdot\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)^\mathsf{T}\right)\mathbf{b}(x_\ell+\varepsilon),
\end{aligned}$$

where σ is defined in (2.24).

We note that a product of the weight function matrix and the basis matrix is

$$\mathbf{W}(x_\ell+\varepsilon)\mathbf{B} = \\
\begin{pmatrix}
b_1(x_1)w_1(x_\ell+\varepsilon) & b_2(x_1)w_1(x_\ell+\varepsilon) & \cdots & b_m(x_1)w_1(x_\ell+\varepsilon)\\
\vdots & \vdots & \ddots & \vdots\\
b_1(x_{\ell-1})w_{\ell-1}(x_\ell+\varepsilon) & b_2(x_{\ell-1})w_{\ell-1}(x_\ell+\varepsilon) & \cdots & b_m(x_{\ell-1})w_{\ell-1}(x_\ell+\varepsilon)\\
b_1(x_\ell)w_\ell(x_\ell+\varepsilon) & b_2(x_\ell)w_\ell(x_\ell+\varepsilon) & \cdots & b_m(x_\ell)w_\ell(x_\ell+\varepsilon)\\
b_1(x_{\ell+1})w_{\ell+1}(x_\ell+\varepsilon) & b_2(x_{\ell+1})w_{\ell+1}(x_\ell+\varepsilon) & \cdots & b_m(x_{\ell+1})w_{\ell+1}(x_\ell+\varepsilon)\\
\vdots & \vdots & \ddots & \vdots\\
b_1(x_N)w_N(x_\ell+\varepsilon) & b_2(x_N)w_N(x_\ell+\varepsilon) & \cdots & b_m(x_N)w_N(x_\ell+\varepsilon)
\end{pmatrix},$$

and hence the equation (2.25) becomes

$$\begin{aligned}
\varphi_\ell(x_\ell+\varepsilon) &= w_\ell(x_\ell+\varepsilon)\Big(\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1} \qquad (2.26)\\
&- \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma}\mathbf{b}^\mathsf{T}(x_\ell)\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)^\mathsf{T}\Big)\mathbf{b}(x_\ell+\varepsilon).
\end{aligned}$$

The central idea is to exploit the Taylor series expansion

$$\mathbf{b}(x_\ell+\varepsilon) = \sum_{i=0}^{\infty} \frac{\varepsilon^i}{i!}\frac{\mathrm{d}}{\mathrm{d}x^i}\mathbf{b}(x_\ell), \qquad (2.27)$$

which is finite if a polynomial basis of a certain order is used.

Expanding the Taylor series we obtain

$$
\begin{aligned}
\varphi_\ell(x_\ell+\varepsilon) &= w_\ell(x_\ell+\varepsilon)\left(\sigma - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma}\cdot\sigma^2\right) \\
&+ \varepsilon w_\ell(x_\ell+\varepsilon)\left(\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell) - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma}\cdot\sigma\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell)\right) \\
&+ \sum_{i=2}^{\infty}\frac{\varepsilon^i}{i!}w_\ell(x_\ell+\varepsilon)\left(\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}^i}{\mathrm{d}x^i}\mathbf{b}(x_\ell) - \frac{1}{\frac{1}{w(\mathbf{x}_\ell+\varepsilon)}+\sigma}\cdot\sigma\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}^i}{\mathrm{d}x^i}\mathbf{b}(x_\ell)\right) \\
&= \frac{w_\ell(x_\ell+\varepsilon)\sigma + w_\ell^2(x_\ell+\varepsilon)\sigma^2 - w_\ell^2(x_\ell+\varepsilon)\sigma^2}{1+w_\ell(x_\ell+\varepsilon)\sigma} \\
&+ \varepsilon\frac{w_\ell(x_\ell+\varepsilon)+w_\ell^2(x_\ell+\varepsilon)\sigma - w_\ell^2(x_\ell+\varepsilon)\sigma}{1+w_\ell(x_\ell+\varepsilon)\sigma}\times \\
&\quad \mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell) + \sum_{i=2}^{\infty}\frac{\varepsilon^i}{i!}\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}^i}{\mathrm{d}x^i}\frac{w_\ell(x_\ell+\varepsilon)}{1+w_\ell(x_\ell+\varepsilon)\sigma}\mathbf{b}(x_\ell) \\
&= \frac{w_\ell(x_\ell+\varepsilon)\sigma}{1+w_\ell(x_\ell+\varepsilon)\sigma} + \varepsilon\frac{w_\ell(x_\ell+\varepsilon)}{1+w_\ell(x_\ell+\varepsilon)\sigma}\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell) \\
&+ \sum_{i=2}^{\infty}\frac{\varepsilon^i}{i!}\frac{w_\ell(x_\ell+\varepsilon)}{1+w_\ell(x_\ell+\varepsilon)\sigma}\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}^i}{\mathrm{d}x^i}\mathbf{b}(x_\ell).
\end{aligned}
$$

Since it holds

$$\lim_{\varepsilon\to 0}\frac{w_\ell(x_\ell+\varepsilon)}{1+w_\ell(x_\ell+\varepsilon)\sigma} = \lim_{\varepsilon\to 0}\frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma} = \frac{1}{\sigma},$$

it immediately follows

$$\varphi_\ell(x_\ell) = 1.$$

Futhermore, due to the fact that

$$\varphi_\ell'(x_\ell) = \lim_{\varepsilon\to 0}\frac{\varphi_\ell(x_\ell+\varepsilon)-\varphi_\ell(x_\ell)}{\varepsilon},$$

we also obtain for the fist derivative of the kernel function

$$\varphi_\ell'(x_\ell) = \frac{1}{\sigma}\mathbf{b}^\mathsf{T}(x_\ell)\mathbf{R}^{-1}\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell),$$

and for the derivatives of higher orders analogously. ☐

Theorem 2.5 *For each node* $\mathbf{x}_\ell$, $\ell = 1, 2, \ldots, N$ *and the index* $k = 1, 2, \ldots, N:\ k \neq \ell$ *it holds for kernel functions and their derivatives*

$$\begin{aligned} \varphi_k(\mathbf{x}_\ell) &= 0, \\ \mathscr{D}_{\mathbf{z}}^{\alpha}\varphi_k(\mathbf{x}_\ell) &= w_k(\mathbf{x}_\ell)\mathbf{b}^T(\mathbf{x}_k)\left(\mathbf{R}^{-1} - \frac{(\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell)) \cdot (\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell))^T}{\mathbf{b}^T(\mathbf{x}_\ell)\mathbf{R}\,\mathbf{b}(\mathbf{x}_\ell)}\right) \cdot \mathscr{D}_{\mathbf{z}}^{\alpha}\mathbf{b}(\mathbf{x}_\ell), \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad k \neq \ell \end{aligned}$$

Proof Without loss of generality we consider the one-dimensional case, i.e. $\mathbf{x}_\ell \equiv x_\ell$ and start with

$$\begin{aligned} \varphi_k(x_\ell + \varepsilon) &= w_k(x_\ell + \varepsilon)\mathbf{b}^\mathsf{T}(x_k) \times \\ & \left(\mathbf{R}^{-1} - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)} + \sigma}\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)^\mathsf{T}\right)\mathbf{b}(x_\ell + \varepsilon), \end{aligned} \tag{2.28}$$

where σ is defined as in (2.24).

Utilizing the Taylor series expansion (2.27) for $\mathbf{b}(x_\ell + \varepsilon)$ yields

$$\begin{aligned} \varphi_k(x_\ell + \varepsilon) &= w_k(x_\ell + \varepsilon) \times \\ & \left(\underbrace{\mathbf{b}^\mathsf{T}(x_k)\mathbf{R}^{-1}\mathbf{b}(x_\ell) - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)} + \sigma}\mathbf{b}^\mathsf{T}(x_k)\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)\underbrace{\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)^\mathsf{T}\mathbf{b}(x_\ell)}_{=\sigma}}_{=\frac{1}{1+w_\ell(x_\ell+\varepsilon)\sigma}\mathbf{b}^\mathsf{T}(x_k)\mathbf{R}^{-1}\mathbf{b}(x_\ell)}\right) \\ &+ \varepsilon w_k(x_\ell + \varepsilon)\mathbf{b}^\mathsf{T}(x_\ell) \times \\ & \left(\underbrace{\mathbf{R}^{-1} - \frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)} + \sigma}\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)\left(\mathbf{R}^{-1}\mathbf{b}(x_\ell)\right)^\mathsf{T}}_{=:\boldsymbol{\Theta}(x_\ell,\varepsilon)}\right)\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell) \\ &+ \sum_{i=2}^{\infty}\frac{\varepsilon^i}{i!}w_k(x_\ell + \varepsilon)\mathbf{b}^\mathsf{T}(x_\ell)\boldsymbol{\Theta}(x_\ell, \varepsilon)\frac{\mathrm{d}^i}{\mathrm{d}x^i}\mathbf{b}(x_\ell). \end{aligned} \tag{2.29}$$

Letting $\varepsilon \to 0$ a finite value of the weight function $w_k(x_\ell)$ can be found, and since it holds

$$\lim_{\varepsilon \to 0}\frac{w_\ell(x_\ell + \varepsilon)}{1 + w_\ell(x_\ell + \varepsilon)\sigma} = \lim_{\varepsilon \to 0}\frac{1}{\frac{1}{w_\ell(x_\ell+\varepsilon)} + \sigma} = \frac{1}{\sigma}$$

and

$$\lim_{\varepsilon\to 0}\frac{w_\ell(x_\ell+\varepsilon)\sigma}{1+w_\ell(x_\ell+\varepsilon)\sigma}=\lim_{\varepsilon\to 0}\frac{\sigma}{\frac{1}{w_\ell(x_\ell+\varepsilon)}+\sigma}=1,$$

it follows

$$\boldsymbol{\Theta}(x_\ell):=\lim_{\varepsilon\to 0}\boldsymbol{\Theta}(x_\ell,\varepsilon)=\mathbf{R}^{-1}-\frac{1}{\sigma}\left(\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)\cdot\left(\mathbf{R}^{-1}\mathbf{b}(\mathbf{x}_\ell)\right)^\mathsf{T}, \tag{2.30}$$

and we obtain

$$\varphi_k(x_\ell)=0.$$

Furthermore, since

$$\varphi_k'(x_\ell)=\lim_{\varepsilon\to 0}\frac{\varphi_k(x_\ell+\varepsilon)-\varphi_k(x_\ell)}{\varepsilon},$$

we get for the first derivatives of the kernel function

$$\varphi_k'(x_\ell)=w_k(x_\ell)\mathbf{b}^\mathsf{T}(x_k)\boldsymbol{\Theta}(x_\ell)\frac{\mathrm{d}}{\mathrm{d}x}\mathbf{b}(x_\ell),$$

and analogously for higher derivatives. □

- ▶ One should notice that the derivatives of the IMLS kernel function, evaluated at the collocation nodes, depend only on the derivatives of the basis functions, which can easily be computed for every node $\mathbf{x}_\ell$. On the other hand, in order to compute the higher order derivatives we must use the higher order basis functions. It must be mentioned that the IMLS kernel functions as defined in (2.10) with weights (2.18) are C^m continuously differentiable functions, where m is the order of the basis $\mathbf{b}$.

Having defined the IMLS kernel functions in (2.10) and the derivatives in Theorems 2.4-2.5, the approximate solution is sought in the form

$$u^{\text{appx}}(\mathbf{x})=\boldsymbol{\varphi}(\mathbf{x})^\mathsf{T}\cdot\mathbf{u}=\sum_{j=1}^{N}\varphi_j(\mathbf{x})\cdot u_j, \tag{2.31}$$

and its derivatives w.r.t. time or space variable $\mathbf{x}=(x_1,x_2,\dots,x_d,t)^\mathsf{T}\in\mathbb{R}^{d+1}$ are

$$\mathscr{D}^\alpha_\mathbf{x}u^{\text{appx}}(\mathbf{x})=\sum_{j=1}^{N}\mathscr{D}^\alpha_\mathbf{x}\varphi_j(\mathbf{x})\cdot u_j=\sum_{i=1}^{N}\sum_{j=1}^{N}\varphi_i(\mathbf{x})\mathscr{D}^\alpha_\mathbf{x}\varphi_i(\mathbf{x}_j)\cdot u_j. \tag{2.32}$$

- ▶ Due to the local support of the weight functions (2.18), the resulting matrices have a sparse structure, since only the neighbouring nodes are contributing to the interpolation.

2.7 Boundary and initial conditions

Applying boundary conditions in meshfree methods has been a problem due to the lack of the Kronecker delta property of the kernel functions, see e.g. [68, 123, 126, 21] . The first approaches which appear in the literature combine the meshfree methods within the domain with finite elements on the boundary to enforce the essential boundary conditions, e.g. [106]. Thereafter many different publications appear aiming to cure this problem, e.g. introducing displacement constraints [211], by a boundary flux method [207], corrected collocations [199], an approach based on d'Alembert's principle [78], Nitsche's and blending technique [60], RKM with nodal interpolating property [31], mixed coordinates and singularities [33], etc.

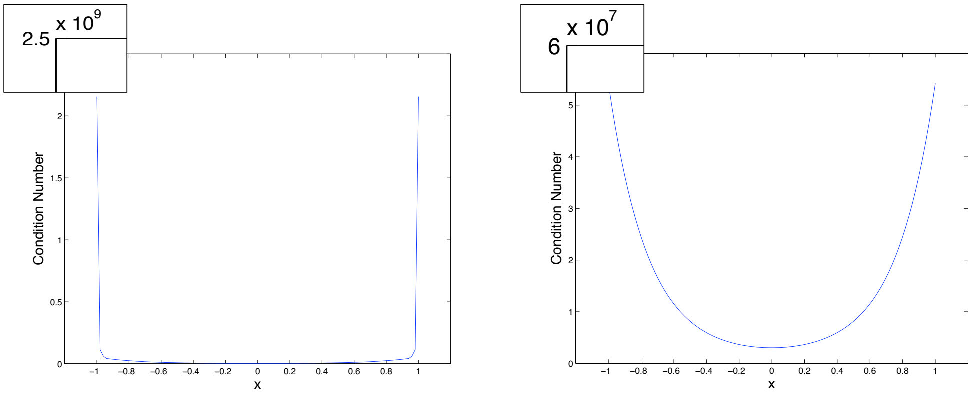
Since the IMLS kernel function do have the Kronecker delta property, the application of boundary conditions is simplified. In this study, however, we focus on the strong-form solution of PDEs, and hence the boundary conditions are applied directly after discretizing the governing equations, as it is done e.g. in the Finite Difference method. The interested reader is referred to the author's papers and articles in conference proceedings [148, 150, 149]. The detailed description of this procedure is provided in Section 6.5 on page 92.

In Section 2.6.2 on page 21 we introduced the strategy to find the inverse of the singular matrix, which occurs in the definition of the kernel functions, in terms of a regular and symmetric matrix **R** defined in the Theorem 2.3 on page 22. However, the IMLS approximation suffers from problems at the boundaries of the computational domain. The reason for this is a violation of some *positivity conditions*, which are described in Section 2.8.2 on page 34. Also some problems occur in the vicinity of the removed singular point, where the weight function may still show very large values. On the other hand, the weights of the points outside of the neighbourhood are zero, since the weight function has a compact support.

As a result the obtained matrix is ill-conditioned. The comparative analysis of the matrices in the middle of the domain and on the boundary for an example of interpolating a function in 1d has shown that the condition number blows up as a computational point is approaching the boundary, see Figure 2.2(a). For numerical studies on accuracy and convergence properties of the proposed method see Chapters 4, 5 and 6.

Since on the boundary the weight function is no longer symmetric in contrast to the situation within the domain, some special treatment is needed. There is also a minimal admissible number of points in the support of the weight function which depends on the highest order of the basis functions used.

The first thing that can be done is to keep the number of points in the support constant. This means that the support of the weight function is increasing as near to the boundary. Another procedure would be to keep the support of the weight function constant throughout the domain. The support has to be large enough to guarantee that even on the boundary there are enough points in the support of the weight function. A better way to treat the boundary is to incorporate some more points outside of the domain into the vector of computational points, namely exactly as many as needed in order to get a fully symmetric weight function



(a) *without 'ghost points'*.

(b) *using 'ghost points'*

Figure 2.2 | Condition number distribution in the computational domain.

for the point on the boundary – just as within the domain. These points are often called 'ghost points' since they serve only to improve the solution behaviour at the boundary.

Figure 2.2(a) shows the distribution of the condition number of matrix **R** from Section 2.6.2 without the special treatment of the boundary, and Figure 2.2(b) – the same distribution but using the 'ghost points'. In this way the problems with the ill-conditioned matrix are "moved to the outside" of the computational domain and the condition number of the matrix is reduced from $2.5 \cdot 10^9$ to $6.0 \cdot 10^7$. Having introduced the 'ghost points', the same SVD-algorithm (see Section 2.6.1 on page 20) can be used on the boundary as within the domain.

▶ For collocation with radial basis functions it is well-known that the error is largest near the boundary [59], and the same holds also for the approach presented in this thesis. Boundary collocation of the PDEs yields an improvement in terms of the accuracy of the method on the boundary if these conditions are introduced as additional equations, herewith increasing the problem size.

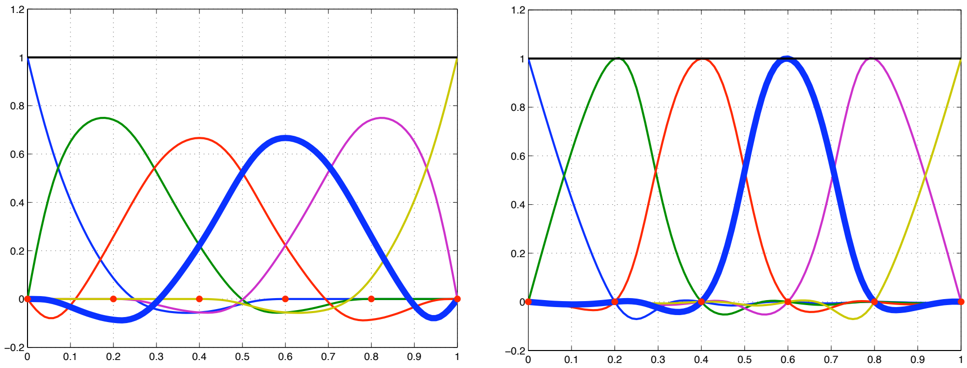
A corrected collocation method for application of essential boundary conditions in meshfree methods is presented in [199], where the authors show the improvement of the convergence rate when using the corrected approach.

2.8 Numerical properties of IMLS functions

In (2.10) we have derived the expression for IMLS kernel functions $\boldsymbol{\varphi}(\mathbf{x})$. Now let us look at their properties.

The IMLS kernel functions are consistent by construction, i.e. the basis functions are represented exactly:

$$\mathbf{B}^{\mathsf{T}} \cdot \boldsymbol{\varphi}(\mathbf{x}) = \mathbf{B}^{\mathsf{T}} \cdot \mathbf{W}(\mathbf{x})\mathbf{B} \left(\mathbf{B}^{\mathsf{T}}\mathbf{W}(\mathbf{x})\mathbf{B}\right)^{-1} \mathbf{b}(\mathbf{x}) = \mathbf{b}(\mathbf{x}). \tag{2.33}$$



(a) *MLS*

(b) *IMLS*

Figure 2.3 | Comparison of MLS and IMLS kernel functions in 1d case for regularly distributed points.

By consistency order n we mean the highest order of polynomials which are represented *exactly* by the method. In terms of kernel functions this reads as

$$\sum_{i=1}^{N} x^q \varphi_i(x) = x^q \qquad \text{for} \quad 0 \leqslant q \leqslant n. \tag{2.34}$$

For a special case of $n = 0$ (the 0^{th} consistency order) we obtain

$$\sum_{i=1}^{N} \varphi_i(x) = 1, \tag{2.35}$$

which means that the kernel functions computed for any point in the domain build a partition of unity.

Figures 2.3(a) and 2.3(b) present the MLS and IMLS kernel functions in 1d, respectively for regularly distributed points in 1d. One can clearly see the difference between the approximating and the interpolating case. The IMLS kernel functions enjoy the Kronecker delta property $\varphi_i(x_j) = \delta_{ij}$, which is not the case for its approximating counterpart – the MLS kernel functions.

It should be pointed out that in both interpolating and approximating case the kernel functions build a partition of unity, represented by black lines in Figures 2.3(a) and 2.3(b). Applying the gradient operator to equation (2.35) we obtain that the sum of the gradients of all kernel functions constitute a partition of nullity:

$$\sum_{i=1}^{N} \nabla \varphi_i(x) = 0. \tag{2.36}$$

2.8.1 Patch tests :: A linear elasticity example

In this section we apply a standard patch test for IMLS kernel functions on an example of linear elastic solids deforming under static loads in two spatial dimensions.

The vector of unknowns is defined for this example as

$$\boldsymbol{\xi} := \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\sigma} \end{pmatrix} = (u_x, u_y, \sigma_{xx}, \sigma_{yy}, \tau_{xy})^\mathsf{T}, \tag{2.37}$$

where $\mathbf{u} = (u_x, u_y)^\mathsf{T}$ are the displacement components in $x-$ and $y-$ direction, respectively and $\boldsymbol{\sigma} = (\sigma_{xx}, \sigma_{yy}, \tau_{xy})^\mathsf{T}$ – the appropriate normal and shear stress components.

Constitutive equations (Hooke's law for 2d elastic solids) giving the stress-strain relation have the following form:

$$\boldsymbol{\sigma} = \mathbf{C}\boldsymbol{\varepsilon} = \mathbf{CDu}, \tag{2.38}$$

where for plane stress the matrix of material constants

$$\mathbf{C} = \frac{E}{1-\nu^2}\begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{pmatrix},$$

and strain components are $\boldsymbol{\varepsilon} = (\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{xy})^\mathsf{T} = \mathbf{Du}$, where ν is the Poisson's ratio, E – the Young modulus and

$$\mathbf{D} := \begin{pmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{pmatrix}. \tag{2.39}$$

For solids in the plane strain state, the matrix $\mathbf{C}$ can be obtained by substituting $E \to \frac{E}{(1-\nu^2)}$ and $\nu \to \frac{\nu}{1-\nu}$.

Equilibrium/governing equations

The equilibrium equations give the relationship between stresses and external forces:

$$\mathbf{D}^\mathsf{T}\boldsymbol{\sigma} + \mathbf{b} = \rho\ddot{\mathbf{u}} + c\dot{\mathbf{u}}, \tag{2.40}$$

where ρ is the mass density, c – the damping coefficient, $\mathbf{b} = (b_x, b_y)^\mathsf{T}$ – the vector of external body forces, $\ddot{\mathbf{u}} = \frac{\partial^2 \mathbf{u}}{\partial t^2}$ – the acceleration, and $\dot{\mathbf{u}} = \frac{\partial \mathbf{u}}{\partial t}$ – the velocity vector.

The patch test is usually restricted to a static one, hence we obtain the equilibrium equation

$$\mathbf{D}^\mathsf{T}\boldsymbol{\sigma} + \mathbf{b} = 0. \tag{2.41}$$

Traction boundary conditions are given in the following form:

$$\begin{pmatrix} n_x & 0 & n_y \\ 0 & n_y & n_x \end{pmatrix} \cdot \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \tau_{xy} \end{pmatrix} = \begin{pmatrix} \bar{t}_x \\ \bar{t}_y \end{pmatrix}. \tag{2.42}$$

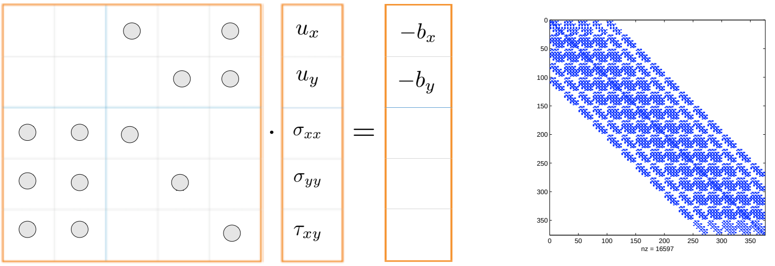
Using a displacement-stress formulation, the system of equation under consideration is

$$\underbrace{\begin{pmatrix} 0_{2\times2} & \mathbf{D}^{\mathsf{T}}_{2\times3} \\ [-\mathbf{CD}]_{3\times2} & \mathbf{I}_{3\times3} \end{pmatrix}}_{:=\mathbf{K}} \cdot \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\sigma} \end{pmatrix} = \underbrace{\begin{pmatrix} -\mathbf{b} \\ 0 \end{pmatrix}}_{:=\mathbf{f}} \tag{2.43}$$

or in a concise form

$$\mathbf{K} \cdot \boldsymbol{\xi} = \mathbf{f}, \tag{2.44}$$

where **K** is the stiffness matrix, and **f** – the right hand side as defined in (2.43).



(a) *Matrix structure for each node* (b) *Stiffness matrix structure*

Figure 2.4 | Matrix structure for patch tests on IMLS kernel functions.

Figure 2.4(a) shows the structure of the discretized stiffness matrix **K** for each of the nodes, where circles represent non-zero entries and Figure 2.4(b) demonstrates the sparse banded stiffness matrix structure after applying boundary conditions. The band width depends on the dilitation parameter of the weight function (2.18) defined in Section 2.5.

▶ A detailed description to discretization of systems of partial differential equations by a proposed approach is given in Chapter 3 on page 35 and the implementation of boundary conditions is extensively described in Section 6.5 on page 92.

Low-order patch test

Material constants chosen for this test are $E = 1.0$, $\nu = 0.25$ and the displacement boundary conditions on all boundaries are prescribed by a linear function.

The criterion to pass this patch test is that stresses and strains are constant, and the displacements within the body are also given by the same linear function.

The STMCM passes this patch test up to the machine accuracy 10^{-16}.

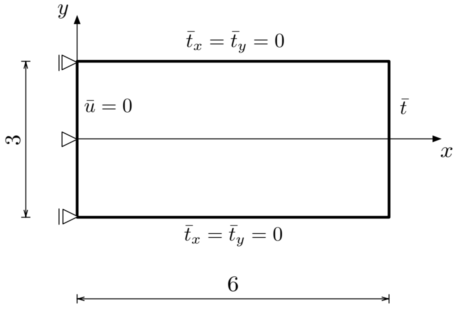


Figure 2.5 | High order patch test.

High-order patch test

In this test we apply a high-order patch test to a beam with material parameters $E = 1.0$, $\nu = 0.25$ for the situation shown in Figure 2.5 for the following boundary conditions:

- top and bottom boundary: $\bar{t}_x = \bar{t}_y = 0$, which implies $\sigma_y = 0, \tau_{xy} = 0$,
- left boundary: $u_x = 0$,
- origin: $u_x = 0$ and $u_y = 0$.

We study two cases of the load applied at the right boundary:

Case 1. linear traction from -1 to 1 as y goes from -1.5 to 1.5:

$\bar{t} = t_x = \frac{2}{3}y$, $t_y = 0$, which implies $\sigma_x = \frac{2}{3}y, \sigma_y = 0, \tau_{xy} = 0$.

The exact solution for displacements in this case is $u_x = \frac{2}{3}xy, \quad u_y = -\frac{1}{3}(x^2 + \frac{y^2}{4})$.

The STMCM passes this high-order patch test with the computational L_2-norm error for both displacements and stresses equal to 10^{-8}.

Case 2. unit traction in x-direction

$\bar{t} = t_x = 1$, $t_y = 0$, which implies $\sigma_x = 1, \sigma_y = 0, \tau_{xy} = 0$

The exact solution for displacements is in this case is also known $u_x = x, \quad u_y = -\frac{y}{4}$.

The STMCM passes this high-order patch test with the computational L_2-norm error for both displacements and stresses equal to 10^{-10}.

Hence, the proposed meshfree collocation method based on IMLS passes the low-order and all high-order patch tests with high accuracy, whereas the conventional collocation method cannot pass the high-order patch test [123], when applied to case 1.

2.8.2 Positivity conditions in meshfree collocations

As mentioned before, the meshfree collocation methods combine the advantages of the meshfree methods (no need to generate a mesh and remesh) with the simplicity of collocations (no need to compute the integrals). However, the robustness of meshfree collocation method has been an issue, especially when used for non-uniform point distributions [67].

If some conditions on the kernel functions and their derivatives are satisfied, the stability of meshfree collocation methods can be assured and the approximation error is bounded. It is shown in [43] that fulfillment of the positivity conditions on the approximation function and its second derivative ensures convergence of finite difference methods with arbitrary irregular meshes for elliptic problems , and in [98] the authors extend the conditions to meshfree methods as

$$\begin{aligned} \varphi_i(\mathbf{x}_j) &\geqslant 0, \qquad i \neq j, && (2.45)\\ \nabla^2\varphi_i(\mathbf{x}_j) &\geqslant 0, \qquad i \neq j, && (2.46)\\ \nabla^2\varphi_i(\mathbf{x}_i) &\leqslant 0. && (2.47) \end{aligned}$$

Patankar postulates the rules in [163] for the construction of finite difference operators and emphasizes that the violation of the positivity conditions leads to physically unrealistic solutions. For a random scattered point distribution the positivity conditions may easily be violated, leading to large errors. Therefore, the meshfree collocation methods are sensitive to the distribution of points and the size of the support of the weight function, especially when the point distribution is highly irregular. We will refer to this issue in Chapter 6.

As proved in [98], the meshfree kernel functions satisfy the first positivity condition (2.45), if and only if they fulfill the Kronecker delta property. This is the case for IMLS kernel functions (see Section 2.6.2 on page 21). However, to satisfy the second positivity condition (2.46), there must be at least one point on either side of the node [98] (for 1d-case), which is obviously not the case on the boundary. In two space dimensions a criterion to check the “quality” of a point cloud is necessary. The author in [180] carries out studies on this matter deriving meshless finite differences for the approximating moving least squares.

Numerical tests showing the importance of point distributions are performed in Chapter 6 on page 73, where, unless stated differently, the ‘ghost-points’ technique is utilized as described in Section 2.7 on page 28 to overcome the violation of positivity conditions. Before we proceed with the numerical studies of various kind of ordinary and partial differential equations in further chapters, let us look in detail at the discretization procedure proposed in this thesis.

"An expert is a man who has made all the mistakes which can be made in a very narrow field."

-- Niels Bohr

"Always listen to experts. They'll tell you what can't be done and why. Then do it!"

-- Robert Heinlein

CHAPTER 3

A Space-Time Meshfree Collocation Method

In this chapter we present the fundamental idea behind the Space-Time Meshfree Collocation Method (STMCM), which is particularly well-suited to the problems with moving boundaries and coupled engineering problems. For this class of problems the established mesh-based methods like FEM suffer from accuracy issues due to distortion of elements and they are more computationally expensive due to the need of remeshing and numerical integration. The usage of a consistent discretization procedure in space as well as in time is known to bring advantages for hyperbolic problems as free-surface and two-fluid flows in the framework of space-time finite elements, see e.g. [105].

This idea is inherited in this thesis, but used in the framework of meshfree collocation methods, where the h-adaptivity can be implemented in a much simpler way compared to the mesh-based methods, see [18, 46, 77], since nodes can be added (or removed) as needed without the costly re-meshing procedure (in this case only a small part of neighbouring kernel functions must be recomputed due to their local character.).

In STMCM the simplicity and straightforwardness of the strong-form computational techniques is combined with advantages of meshfree methods and a consistent approximation technique is used for both space and time discretizations.

3.1 Splitting into space-time slabs

For the sake of simplicity let us look at a three-dimensional (two space and one time directions, i.e. $\mathbf{x} := (x_1, x_2, t)^\mathsf{T}$) initial boundary value problem (IBVP) in a space-time domain $Q = \Omega \times I$ with the boundary Γ which is constant in time:

$$\mathscr{L}\,\mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \qquad \text{in} \quad Q, \tag{3.1}$$

$$\mathscr{B}\,\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) \qquad \text{on} \quad \Gamma, \tag{3.2}$$

$$\mathbf{u}(\mathbf{x}_0) = \mathbf{u}_0, \tag{3.3}$$

where $\mathscr{L}$ is the differential operator containing partial derivatives w.r.t. space and time, $\mathscr{B}$ is a boundary (differential) operator, and $\mathbf{u}_0$ is the initial condition at time t_0. The problem is defined for the time interval $I = [0, T]$ and the computational domain of interest Ω consists of two parts, i.e. $\Omega = \Omega_a \cup \Omega_i$ with an interface Σ between the two subdomains. The physical meaning of these subdomains will become clear in further sections.

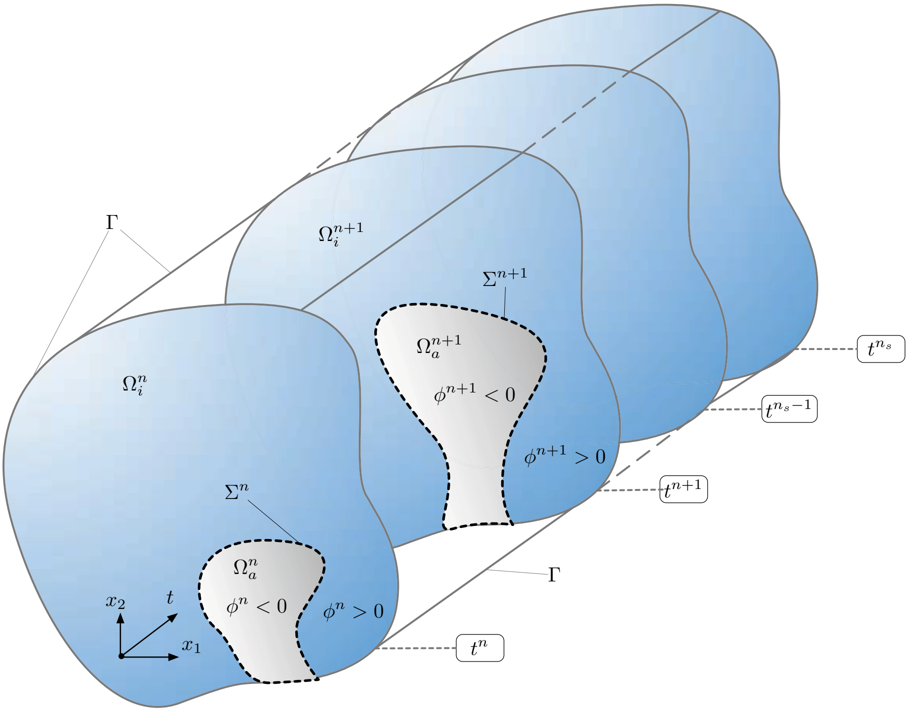


Figure 3.1 | Splitting of the space-time domain into time slabs.

We divide the whole space-time domain into n_s time slabs $\left[t^{(n)}; t^{(n+1)}\right]$, $n = 1 \ldots n_s$ as shown on Figure 3.1, and compute the solution $\mathbf{u}(\mathbf{x})$ successively for each of the time slabs. The outer boundary Γ is of arbitrary shape which is assumed to be constant in time.

Here ϕ^n is the level-set function at time t^n, which is used as initial condition for the solution of the level-set equation. As a result one obtains a value of the level-set function ϕ^{n+1} at the end of the time slab for time t^{n+1}. This value is then used again as the initial condition for the next time slab. The domains Ω_a and Ω_i, as well as the interface Σ, which are moving and changing its topology, are denoted for different time instants with the indices n and $n+1$ respectively.

- ▶ It should be pointed out that the time slabs can be, in general, of different sizes, and so are the time and space steps Δt and Δx, respectively. Also it should be noted that the solution is computed not only at the beginning and at the end of time slabs, but also in between. So we deal not with a *time*-stepping, but rather with a *time-slab*-stepping procedure.

In the presented Space-Time Meshfree Collocation Method initial conditions can also be understood as boundary conditions. We will name them differently, keeping in mind that the initial conditions are boundary conditions on the boundary with $t = 0$. One of the advantages of this fully implicit time discretization method is the adaptivity in both space and time. Since a mesh is present neither for nodal values, nor for integration, one can easily adapt simply by adding (or removing) nodes as needed. Moreover, since the consistent discretization is used for both space and time, the adaptivity process is applicable in space as well as in time. Of course in this case the kernel functions must be recomputed in a small neighbourhood of the inserted (or removed) nodes (see Section 3.4.2 on page 40).

3.2 Discretization of a coupled problem

In this section we apply the technique presented in Section 2.2 on page 12 to the mathematical model for biofilm growth described later in Chapter 7 on page 131. The physical meaning of the subdomains aren't decisive at this stage, they will become apparent in Chapter 7. Two subdomains and the interface between them, which is moving and evolving in time, are considered. For the sake of simplicity in the following we describe a two-dimensional example (one dimension for space x and one for time t).

Figure 3.2 shows qualitatively a typical space-time slab discretization by a set of N (randomly distributed) points. One part of the domain Ω_a^n at the time level t^n has an interface Σ^n with another one Ω_i^n. After the solution for the time slab is found, one will obtain a new domain distribution at the time level t^{n+1} which will be the initial condition for the next time slab. The dashed line in Figure 3.2 represents the evolution of the interface $\Sigma(t)$ between the two subdomains (e.g. formation of the biomass in the context of biofilm growth, see Chapter 7), and it is implicitely given by the zero-level of the level-set function ϕ.

Derivatives of the kernel functions must be computed only once for a given set of nodes, and in the case of added (removed) nodes, the kernel functions and its derivatives are recomputed only in the small neighbourhood of the added (removed) nodes due to the local support of the weight function (2.18).

As already mentioned in Section 2.2 on page 12, the choice of the basis needed for computing the kernel functions and its derivatives depends on the problem under consideration as

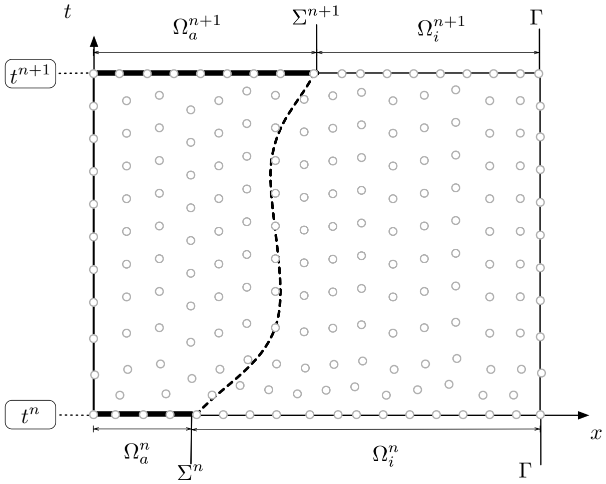


Figure 3.2 | Discretization of the space-time slab.

well as on the desired convergence order of the solution. Monomials of a certain order are frequently used. In STMCM we may choose between different or the same orders of space and time discretizations. For example if we use a linear basis in time and a quadratic in space, the basis vector is $\mathbf{b}(x,t) = (1, x, t, x^2)^\mathsf{T}$, and $\mathbf{b}(x_1, x_2, t) = (1, x_1, x_2, t, x_1^2, x_1 x_2, x_2^2)^\mathsf{T}$ for the problems in one and two space dimensions, respectively.

3.3 Strategy for nonlinearities

Mathematical models describing behaviour of engineering problems or physical phenomena will usually have nonlinearities (e.g. due to material or geometry nonlinearities, large deformations, nonlinear boundary conditions, coupled problems, etc.). In any case the general task is to solve a nonlinear system of the form $\mathbf{F}(\mathbf{u}) = 0$.

A wide range of iterative methods to solve this nonlinear system exists, e.g. Newton, quasi-Newton, modified Newton, Newton-Raphson methods, conjugate gradients, etc., see e.g. [139, 216, 206] and references therein.

In STMCM a nonlinear system, which is to be solved for each time slab, can be written in a generic matrix-vector notation as

$$\mathbf{A}(\mathbf{u}) \cdot \mathbf{u} = \mathbf{b}(\mathbf{u}),$$

where $\mathbf{A}$ is the system matrix for a single space-time slab, $\mathbf{b}$ is the right-hand side, and $\mathbf{u}$ is the solution of the whole slab. In our computations the system is linearized and nonlinearities are resolved via the fixed point iteration scheme (Picard iteration) as

$$\mathbf{A}(\mathbf{u}^{\mathtt{i}-1}) \cdot \mathbf{u}^{\mathtt{i}} = \mathbf{b}(\mathbf{u}^{\mathtt{i}-1}),$$

where `i` is the iteration number. Although this technique is only linearly convergent, it is simple yet powerful and gives good results without the need of computing the Jacobian matrix as in the case of Newton-like methods and its modifications.

3.4 Point distribution techniques

In this section we will briefly discuss the techniques used to distribute the points in a computational domain, as well as a search technique used to determine the points lying in the support domain of the weight function, which is essential for construction of the kernel IMLS functions (2.10) as described in Section 2.2 on page 12.

Efficiency of the algorithms for numerical solution of partial differential equations depends strongly on the underlying data structure. A good overview of the possibilites to analyze and design data structure is given in [175, 203], multiresolution methods in the modelling of scattered data are studied extensively in [96], and fast and effective algorithms for finding neighbouring points can be found in [179].

3.4.1 Halton points

Halton points [80], which are often used in quasi Monte Carlo methods for numerical integration in multiple dimensions, are constructed from van der Corput sequences [205]. We will use a multidimensional Halton sequence (defined below) to construct a *well-distributed* pattern of points in the computational domain.

Let us consider the following lemma.

Lemma 3.1 *Every non-negative integer n can be uniquely represented using a prime basis p and integer coefficients x_i, $i = 0, \dots, k$ as*

$$n = \sum_{i=0}^{k} x_i p^i, \qquad 0 \le x_i < p.$$

Proof The proof is beyond the scope of this thesis, see [80] for details. We just notice that e.g. if $n = 9$, then

$$9 = 0 \cdot 3^0 + 0 \cdot 3^1 + 1 \cdot 3^2,$$

with $p = 3$, $k = 2$, the coefficients $x_1 = x_2 = 0$, and $x_3 = 1$. □

We can now define a van der Corput sequence needed for construction of Halton points.

Definition 3.2 *A sequence $h_{p,N} = \{h_p(n) : \ n = 0, 1, 2, \dots, N\}$ with functions $h_p(n) : n \rightarrow [0, 1)$ defined as*

$$h_p(n) = \sum_{i=0}^{k} \frac{x_i}{p^{i+1}}$$

is called a van der Corput sequence.

For instance for $p=3$ *and* $N=9$ *:* $\quad h_{3,9}=\left\{0,\frac{1}{3},\frac{2}{3},\frac{1}{9},\frac{4}{9},\frac{7}{9},\frac{2}{9},\frac{5}{9},\frac{8}{9},\frac{1}{27}\right\}.$

And finally, we are ready to define a set of Halton points.

Definition 3.3 *A* Halton points set $\mathsf{H}_{d,N}$ *of* $N+1$ *Halton points in a* d*-dimensional unit cube* $[0,1)^d$ *is defined as a set of* d *van der Corput sequences for (distinct) primes* $p_1,\ldots,p_d$*, i.e.*

$$\mathsf{H}_{d,N}:=\{h_{p_1}(n),\ldots,h_{p_d}(n):\ n=0,1,2,\ldots,N\}.$$

Figure 3.3 on the facing page and Figure 3.4 on page 42 show examples of Halton points distribution for $d=2$ and $N=1024$ on a unit square and unit circle, respectively.

An important and useful property of Halton points, which distinguishes them from the Hammersley points [59], is that the Halton point sets can be constructed *sequentially*, since these are *nested* point sets, i.e.

$$\mathsf{H}_{d,M}\subset\mathsf{H}_{d,N}\quad\text{for}\quad M<N,$$

and therefore, one does not need to start from scratch when extending an existing set with some additional points during an adaptive refinement.

3.4.2 Data management

An efficient data management is important when dealing with large data sets. When constructing the IMLS kernel functions (see Sections 2.2 and 2.5) we often need to know which data points belong to the support of a given weight function. Moreover, we are strongly interested in finding the m nearest neighbours of a given point.

These tasks are carried out in this thesis via *d-dimensional trees* (d-d trees) – a space-time-partitioning data structure for organizing points in a d-dimensional space. These trees developed in [23] are a substantial improvement over the classical quad- and octtrees, since they allow a *dimension-independent* implementation which will be of use when applying the proposed methods to higher-dimensional problems. The idea of a d-d tree, often called in literature 'k-d tree', is to break down a set of N data points in $\mathbb{R}^d$ hierarchically into a number of subsets (relatively small), containing about the same number of data points each.

Figure 3.5 on page 43 explains briefly the standard d-d tree procedure. Each tree node is obtained via splitting the domain by a plane which passes through a data point and is orthogonal to one of the coordinate axes. The built subsets, children, contain about the half of the parent node points. The children are then partitioned into equal halves via partitioning planes, usually cycling through a different dimension. Other possible splitting procedures, including adaptive d-d trees, i.e. a tree where successive levels are split along different dimensions, are described in [203].

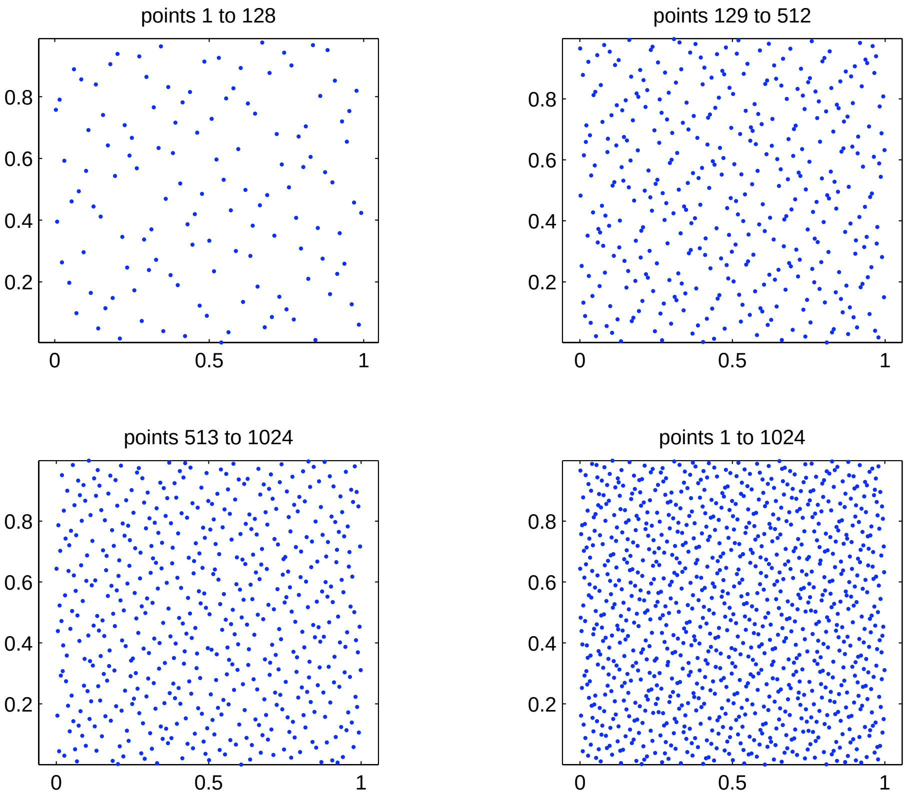


Figure 3.3 | Halton points on a unit square.

The computational complexity to build a d-d tree using the standard procedure out of N points in $\mathbb{R}^d$ is

$$\mathsf{t}_{\mathsf{tree}} = \mathcal{O}(d \cdot N \log N)$$

and it requires

$$\mathsf{s}_{\mathsf{tree}} = \mathcal{O}(dN)$$

space.

After the tree is built, any of the queries needed to construct the IMLS kernel functions and their derivatives, including inserting new points or removing them as well as defining the nearest neighbours to an arbitrary point in the tree, is executed in

$$\mathsf{t}_{\mathsf{query}} = \mathcal{O}(\log N)$$

time. Without using the d-d tree the computational complexity would have been $\mathcal{O}(N)$. Furthermore, the d-d tree strategy is applicable to three and higher dimensions in an absolutely analogous way to the procedure described above for two dimensions.

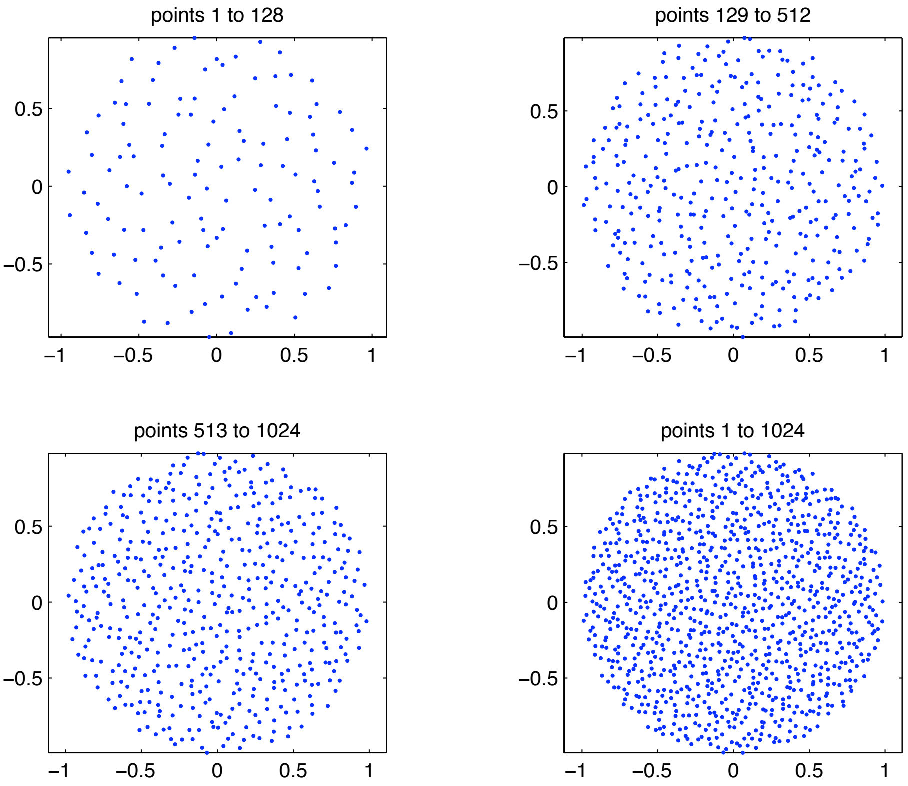


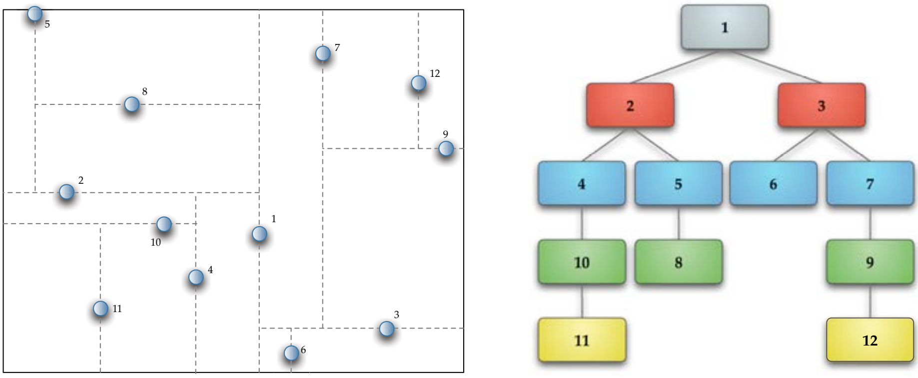
Figure 3.4 | Halton points on a unit circle.

- Among other possibilities a so called *hash grid* is often used. This technique maps an arbitrary point $\mathbf{x} = (x,y,z)^\mathsf{T}$ lying in a 3d cell of size L onto a cell of a uniform grid with index k as follows

$$\mathbf{x} = (x,y,z)^\mathsf{T} \longmapsto k = \mathsf{h}(\lfloor x/L \rfloor, \lfloor y/L \rfloor, \lfloor z/L \rfloor), \tag{3.4}$$

where the hash function h rounds down the scaled coordinates to the next integer. A single point is inserted into the hash grid in $\mathcal{O}(1)$, and hence for N computational points the grid is linearly build in $\mathcal{O}(N)$ time, and so is the computational complexity for a range query using a hash grid. Therefore, especially for a high number of computational points N, the d-d tree data management technique is advantageous since in the latter case any query is performed in $\mathcal{O}(\log N)$ time.

- Since the Halton points have a property to be distributed within the domain and not on its boundary, the final set of reference Halton data points is augmented with a set of uniformly distributed points on the boundary of the computational domain under consideration. The geometry of the domain is usually given for a specific problem in form of e.g. CAD data.



(a) *d-d partitioning* (b) *d-d tree*

Figure 3.5 | *d*-d partitioning and a *d*-d tree for 12 Halton points.

- A technique for distributing computational points within domains with irregular boundaries, implicitly described by a zero-isocontour of a higher-order hypersurface, is the subject of Chapter 4 and its results are summarized by the author in [154].

3.5 Numerical stability and trade-off principles

Halton points are well-distributed random points. For a set of Halton points $\mathscr{S}$ a so-called *fill distance* (in literature sometimes also called ‘covering radius’), denoting how well the set $\mathscr{S}$ fills out (or covers) the computational domain of interest Ω, is defined

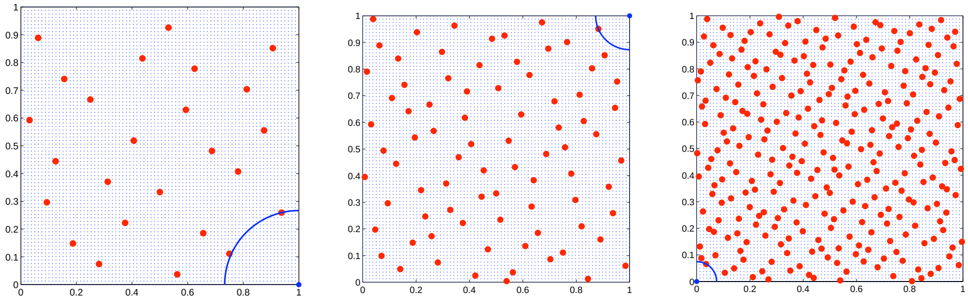
$$h_{\Omega,\mathscr{S}} := \sup_{\mathbf{x}\in\Omega} \min_{\mathbf{x}_i\in\mathscr{S}} \|\mathbf{x}-\mathbf{x}_i\|_2 . \tag{3.5}$$

The geometrical interpretation of the fill distance is the radius of the largest ball which can ‘fall through’ the reference data set. Figures 3.6 on the next page shows exemplarily the fill distances for sets of 25, 81, and 289 Halton points.

A typical support of the weight function (2.18) assigned to some point $\mathbf{x}\in\Omega$ for a set of $N = 289$ Halton points is shown in Figure 3.7 on the following page, while the dilatation parameter being kept constant throughout the domain at the value $\rho = 2\cdot h_{\Omega,\mathscr{S}}$.

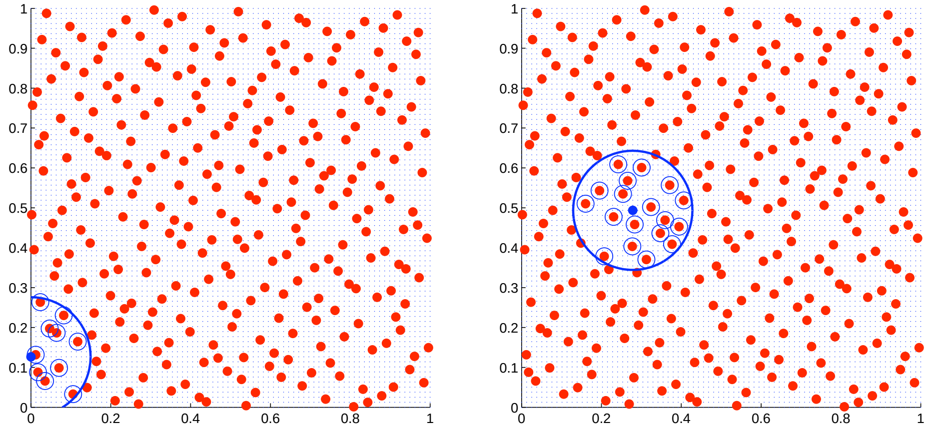
3.5.1 Choice of parameters

In the presented STMCM there is a set of parameters that are to be chosen, which is often referred to as a ‘trade-off principle’ in the context of meshfree methods. This section provides the reader with an overview and guidelines on choosing these parameters based on



(a) $N = 25 \mid h_{\Omega,\mathscr{S}} \approx 0,267$ (b) $N = 81 \mid h_{\Omega,\mathscr{S}} \approx 0,127$ (c) $N = 289 \mid h_{\Omega,\mathscr{S}} \approx 0,075$

Figure 3.6 | Fill distances $h_{\Omega,\mathscr{S}}$ for sets of N Halton points.



(a) *on a boundary* (b) *within the domain*

Figure 3.7 | A typical support of a weight function with a dilatation parameter $\rho = 2 \cdot h_{\Omega,\mathscr{S}}$ for a set of $N = 289$ Halton points in a unit square.

own numerical experiments (see Chapters 4 – 6) as well as recommendations from literature (e.g. [123, 122, 124]).

One can subdivide the whole set of parameters of the STMCM into two parts:

Part I | Interpolating Moving Least Squares parameters

- q: order of the basis (see Section 2.4).

  The choice of the order of the basis functions depends in our approach on the highest order of derivatives of governing equations. In our numerical experiments we will concentrate upon linear and quadratic cases, i.e. $q = 1$ and $q = 2$, respectively.

- α: exponent of the singular weight function (see Section 2.5).

  The parameter α of the weight function affects the ratio of influence of the neighbouring points lying further away to those lying closer to the point of consideration. We suggest using the values $\alpha = 4, 6, 8$. These values will assure the interpolative property of the method, without blowing the condition number of the matrix that is to be inverted when constructing the IMLS kernel functions and their derivatives. At the same time, choosing higher values of α prevents the points lying further away from getting too much weight.

- ρ: dilatation parameter, also called 'domain of influence' . (see Section 2.5).

  The dilatation parameter is a core parameter of the method. The 'rule of thumb' for its choice is: it must be as small as possible, but large enough to avoid spurious short-wave-oscillations in the numerical solution (so called *aliasing*) and so that the whole computational domain be covered by supports of the weight functions.
  The values for the dilatation parameters based on own experiments and using recommendations from the literature, e.g.[123, 59] are $\rho = (2 \div 3.5) \cdot h_{\Omega,\mathscr{S}}$, whereas the coefficient in front of the fill distance $h_{\Omega,\mathscr{S}}$ depends on the order of the basis q: higher-order-basis need more neighbouring points; too large values of ρ, however, lead to excessive smoothing [93].

Part II | Space-Time parameters

- δ_n: thickness of a time slab.

  The thickness of a time slab is analogous to a time step in conventional time integration methods.

- N_n: number of points to discretize a time slab.

  To discretize small time slabs we use the minimum N_n points needed by the basis (a onestep-like time integration method), and if δ_n is chosen bigger, the number N_n must also be increased (a multistep-like time integration method).
  In Section 6.3.2 on page 82 we compare the onestep-like and multistep-like time integration methods depending on choice of δ_n and N_n.

3.6 Workflow

The workflow of mathematical modelling by STMCM can be summarized into the following steps:

1. Consider a mathematical model consisting of a system of nonlinear time-dependent PDEs defined in a computational space-time domain $Q = \Omega \times I$ together with boundary and initial conditions on Γ.
2. Split the space-time domain Q into time slabs $Q^n = \Omega \times I^n$ with time intervals I^n.
3. Discretize a time slab Q^n:
    (a) Distribute computational points.

    Halton points distribution are used as described in Section 3.4 on page 39. A specific procedure required for irregularly-shaped domains, especially in 3d, is a subject of Chapter 4.

    (b) Compute kernel functions and their derivatives.

    The formulae from Section 2.6.3 on page 23 are used.

    (c) Resolve nonlinearities.

    The strategy presented in Section 3.3 on page 38 is used.

    (d) Discretize the underlying PDEs in strong form and apply boundary and initial conditions.

    According to description from Section 2.7 on page 28.
4. Obtain the solution for the current time slab.
5. Proceed to the next time slab $n = n + 1$ and goto Step 3.

- ► In the case the distributed points do not change throughout the computational process, the kernel functions and their derivatives needed for discretization of the strong form of underlying PDEs are computed only once (!) in advance and are reused for each of the time slabs, i.e. the steps 3(a) and 3(b) are taken out of the loop. Therefore, in this case the computational cost is significantly reduced.

- ► If the problem under consideration is not time-dependent, the STMCM reduces to MCM in space, and if the method is applied to ODEs, then the method becomes MCM in time.

"The true delight is in the finding out rather than in the knowing."

-- Isaac Asimov

CHAPTER 4

Discretization of Irregularly-Shaped Domains

The distribution of computational points in a domain of interest, which is a part of a workflow of any meshfree method, is not a trivial task especially for irregularly-shaped domains in three spatial dimensions and a specific technique is required for this purpose.

One of the advantageous features of Halton points, as described in Section 3.4.2 on page 40, is the easiness and applicability to multi-dimensions. However, in s-dimensions these points are defined on a unit volume $[0,1)^s$ only (unit square for 2d, or unit cube for 3d). Therefore, in order to be able to use the Halton points for domains with complex boundaries one needs some strategy to 'map' them onto the irregular domain shape.

The aim of this chapter is the development of an implicit strategy to 'map' the Halton points onto a domain of irregular shape via scattered data interpolation. The idea was inspired by the level set method [162]. Since the domain of interest is as a rule given in form of CAD data, these data can be interpolated and represented by a zero-level of a higher-dimensional function (the level-set function) which, therefore, has different signs inside and outside of the domain. Subsequently, a set of Halton points scaled to the bounding box of the CAD data can be generated and the Halton points lying within the computational domain can be easily extracted using a Heaviside function, i.e. by looking at the sign of the level-set function.

4.1 Scattered data interpolation

Let us briefly recall at this stage the idea of the Interpolating Moving Least Squares method (see Section 2.2 on page 12 for a detailed derivation) from an interpolation viewpoint.

Given a set of s-dimensional reference data points $\mathscr{S} : \{\mathbf{x}_i \in \mathbb{R}^s,\ i = 1,\ldots,N\}$ and corresponding data values at these points $z_i \in \mathbb{R}$, the values of an interpolating function at a specific point $\mathbf{x}_\ell \in \mathbb{R}^s$ can be computed as

$$z(\mathbf{x}_\ell) = \sum_{i=1}^{N} \varphi_i(\mathbf{x}_\ell) \cdot z_i,$$

where $\varphi_i(\mathbf{x})$ are the Interpolating Moving Least Squares functions defined in (2.10). It must be emphasized that the IMLS functions are, as their name implies, *interpolating* the data, i.e. if $\mathbf{x}_\ell \equiv \mathbf{x}_i \in \mathscr{S}$ then $\varphi_i(\mathbf{x}_j) = \delta_{ij} \implies z(\mathbf{x}_i) = z_i$, which is not the case for traditional (*approximating*) MLS, are utilized. Keeping in mind that our goal is to solve ordinary and partial differential equations, an interpolatory method is preferable (see Chapter 6 on page 73). However, e.g. in computer graphics applications, if data contain noise then the approximating MLS approach should be used, since approximating surfaces smooth out the reference data.

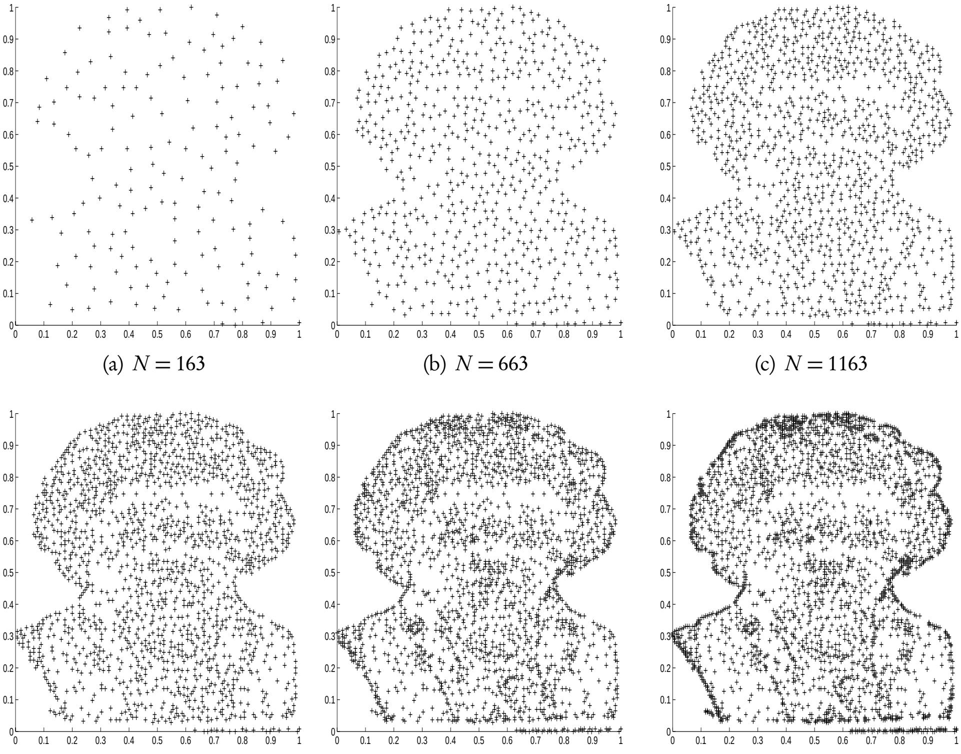
The fact that reference data points $\mathbf{x}_i$ lie in an arbitrary s-dimensional space enables the problem of scattered data interpolation to cover various applications. If $s = 1$ and the data values z_i are some measurements made at certain times, then the reference points x_i would correspond to different instants in time. If, for instance, we want to produce a relief map of the ground surface in deep sea waters, then $\mathbf{x}_i \in \mathbb{R}^2$ ($s = 2$) would correspond to the two coordinates in the plane, and the data values z_i – to the appropriate depth of the sea. For the case $s = 3$ one could be interested in a temperature distribution inside of a building, a solid body, etc., or the data comes from digitizing surfaces of sculptures, machinery, medical or archaeological objects. Problems for higher dimensions exist in optimization, finance, statistics, geological modelling, etc.

4.2 A multilevel surface reconstruction

We start with a qualitative example of a multilevel surface reconstruction based on a data set for a digitized surface of a bust of the German composer Ludwig van Beethoven, consisting of 2663 points on a unit square. The data for this example were taken from [59].

Using the thinning algorithm for scattered data [96], a nested series of datasets $\{\mathscr{S}_k\}$, $k = 1,\dots,6$ with 163, 663, 1163, 1663, 2163, and 2663 reference points respectively is constructed, see Figure 4.1 on the next page. The approximated values of the bust surface are evaluated on a regular 80×80 regular grid. Since no analytical formula of the surface describing Beethoven's face exists, we can only perform a qualitative analysis of the reconstructed surface. Since in this example we are only interested in reconstructing the surface, i.e. in the values of the function itself and not its derivatives, it is sufficient to use a complete linear basis function $\mathbf{b}(\mathbf{x}) = \mathbf{b}(x_1, x_2) = (1, x_1, x_2)^\mathsf{T}$ to compute the IMLS kernel functions.

Figures 3.6 on page 44 and 4.2 on page 50 show the fill distances for sets of Halton points and of the digitized data of the Beethoven's bust respectively. One can clearly see the different behaviour when computing the fill distance measure for these two point arrangements. Whereas for an increasing number of Halton points the fill distance decreases, it remains



(a) $N = 163$ (b) $N = 663$ (c) $N = 1163$

(d) $N = 1663$ (e) $N = 2163$ (f) $N = 2663$

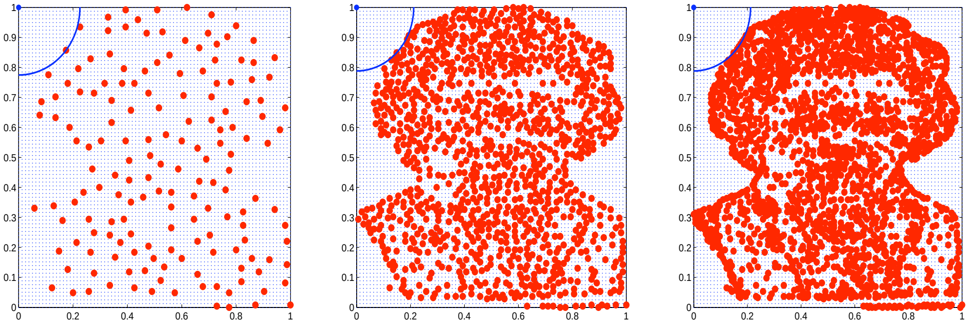
Figure 4.1 | A series of nested datasets with N reference points of the Beethoven's bust.

basically the same when computed for nested subsets of Beethoven's data due to the non-uniformity of the available data in the latter case.

If the fill distance is computed as defined in (3.5) for the whole computational domain and is kept constant, one speaks of a *stationary* interpolation or approximation. This approach is well suitable for uniformly, randomly distributed sets of reference data, e.g. Halton points. If the reference data is not uniformly distributed, i.e. denser in one part of the computational domain and has gaps in the other, then a more sophisticated *non-stationary* interpolation or approximation with varying fill distance throughout the domain can be implemented.

The multilevel stationary reconstructions of Beethoven's bust data for each of the datasets $\{\mathcal{S}_k\}$ are shown from the top view in Figure 4.3 on page 51, and a side view of the reconstructed surface for $N = 2663$ reference points is presented in Figure 4.4 on page 52. One can clearly see the expected behaviour, i.e. as the number of reference data points increases, the resolution of the relief details of the bust surface becomes qualitatively better.

The parameters of the weight function (2.18), defined in Section 2.5 on page 17, were chosen



(a) $N = 163 \mid h_{\Omega,\mathscr{S}} \approx 0,225$ (b) $N = 1163 \mid h_{\Omega,\mathscr{S}} \approx 0,212$ (c) $N = 2163 \mid h_{\Omega,\mathscr{S}} \approx 0,212$

Figure 4.2 | Fill distances $h_{\Omega,\mathscr{S}}$ for Beethoven's bust data sets.

for this example $\alpha = 6.0$, and the dilatation parameter – according to recommendations in [123] as $\rho = 2 \cdot h_{\Omega,\mathscr{S}}$, where $h_{\Omega,\mathscr{S}}$ is the fill distance as defined in (3.5) in Section 3.5 on page 43.

Figure 4.5 on page 53 demonstrates the influence of the parameter α of the weight function on the quality of the reconstruction. For smaller parameters α the quality of the reconstructed surface is very poor, since the farther points obtain too much weight. Consequently, bumps and dimples are produced. Increasing α the influence ratio of nearest reference points to the points further away from the point of interest is increased, and the recovery is getting better.

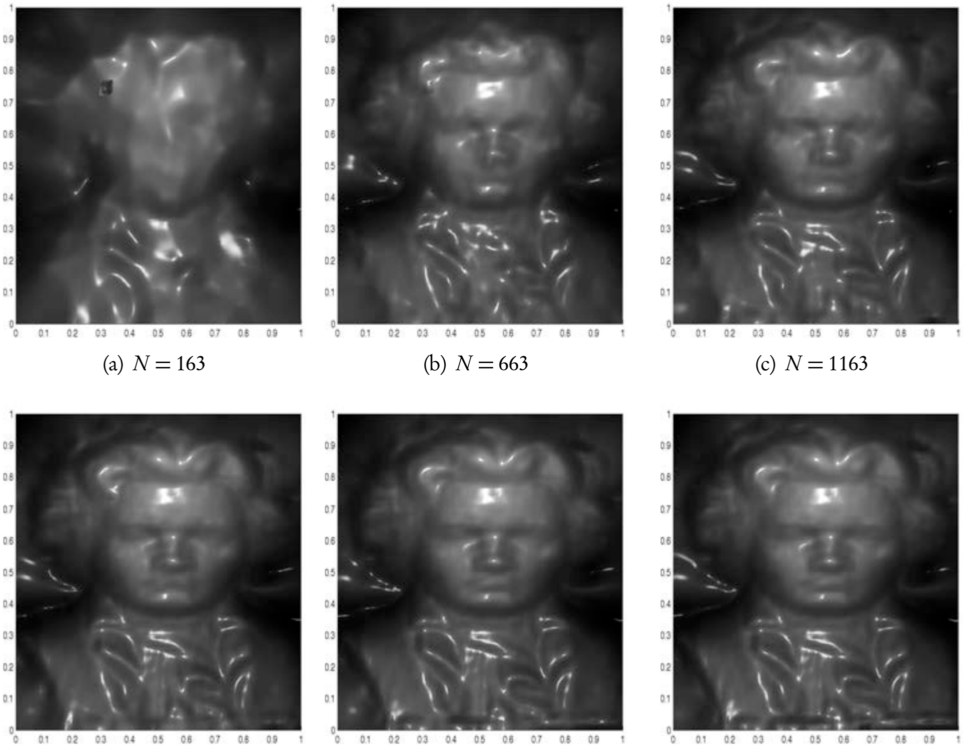
We will study the parameter dependency of the solution in further sections, where we perform a convergence study while interpolating functions with given analytical formulae and computing their derivatives (see Chapter 5 on page 63).

4.3 Interpolation of point cloud data by implicit surfaces

A reconstruction of *implicit* surfaces is a demanding task in such areas as computer aided design (CAD), computer graphics, medical imaging, reverse engineering, etc., where one often deals with huge amounts of irregular points in 3d – the *point cloud data*. In such cases a surface under consideration can no longer be explicitly represented.

Given the three-dimensional data set $\mathscr{S} : \{\mathbf{x}_i \in \mathbb{R}^3,\ i = 1, \ldots, N\}$, which comes from a two-dimensional manifold $\mathscr{M}$, we search for a surface $\hat{\mathscr{M}}$ that is an interpolation of $\mathscr{M}$. The two-dimensional manifold $\mathscr{M}$ is defined implicitly by a zero-isosurface of some function $\phi(\mathbf{x})$ as

$$\mathscr{M} : \{\mathbf{x} \in \mathbb{R}^3 : \phi(x_1, x_2, x_3) = 0\}.$$



(a) $N = 163$ (b) $N = 663$ (c) $N = 1163$

(d) $N = 1663$ (e) $N = 2163$ (f) $N = 2663$

Figure 4.3 | A stationary reconstruction of Beethoven's bust by IMLS using N reference points. Approximated values of the reconstructed surface were evaluated on a regular 80×80 grid for visualization.

The normal of the manifold $\mathscr{M}$ is defined by gradient of ϕ wherever the reconstruction is smooth enough.

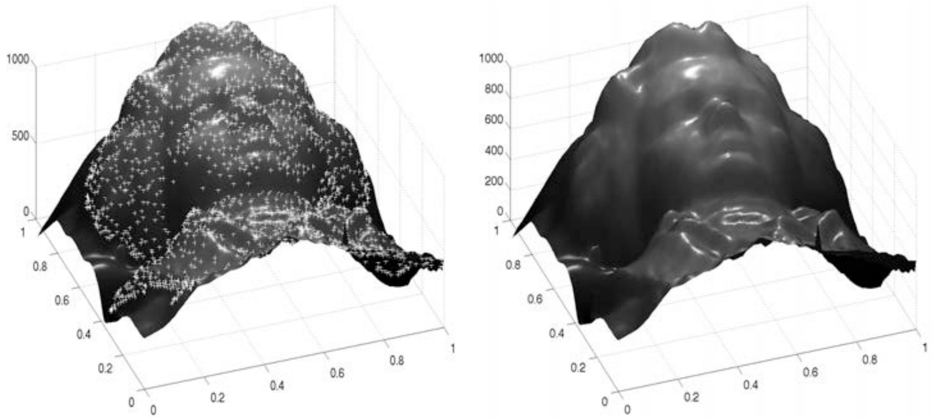
A substantial difference between the previous example and the interpolation by implicit surfaces must be noted. In the latter case a condition on the interpolant to be zero at data points is *not sufficient* to reconstruct the whole surface, since, for instance, a zero-function satisfies it. Hence, the problem is not well-posed. In order to approximate the trivariate function ϕ from the given reference data set $\mathscr{S}$, it must be "extended", i.e. some additional *off-surface* points must be added, see e.g. [203, 59].

One usually adds the points $\tilde{\mathbf{x}}$ lying *inside* and *outside* the manifold $\mathscr{M}$ stepping a small distance δ from the surface in the direction of the inner and outer normal vector, respectively:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i \pm \delta \mathbf{n}_i, \quad i = 1 \ldots N, \tag{4.1}$$

where $\mathbf{n}_i$ is the outer normal of the appropriate reference data point $\mathbf{x}_i \in \mathscr{S}$.

Other procedures on how to extend the data are possible, e.g. propagation methods like the



(a) *reference points on reconstructed surface* (b) *reconstructed surface*

Figure 4.4 | Side view at the reconstructed surface of Beethoven's bust using $N = 2663$ reference points. Approximated values of the reconstructed surface were evaluated on a regular 80×80 grid.

level-set and fast marching techniques [181], or hyperbolic grid generation (see [147] and references therein).

- If data stem from a digitized procedure by a laser scanner, the normals are usually not available and they must be computed from the point cloud data set itself. A number of techniques for this purpose has been developed, see e.g. [203].

Let us assume that the normal vectors for each of the reference data points are available, and hence we may "extend" the data according to (4.1). Once the point cloud data is "extended", the interpolating function is determined in such a way that its zero-isocontour interpolates the original point cloud data set, and some (arbitrary, but with contrary signs) values $\tilde{\phi}$ are interpolated at the "outer" and "inner" extended data points.

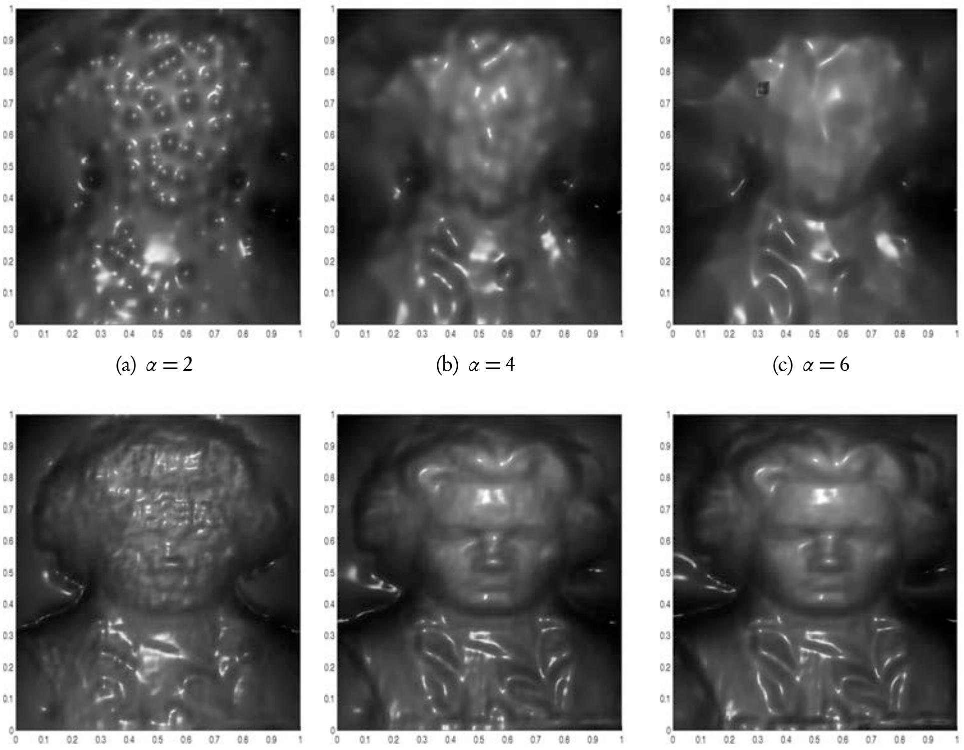
$$\phi(\mathbf{x}_i) = 0, \tag{4.2}$$

$$\phi(\tilde{\mathbf{x}}_i) = \pm\tilde{\phi}. \tag{4.3}$$

The values $\tilde{\phi}$ for extended data points are arbitrary. The choice of δ parameter in (4.1) in contrast is important and it must be chosen with care since it might be critical for a good surface fit [29]. Especially, if δ is too large, it may lead to self-intersecting surfaces as we shall see later. As suggested in [59], we will use δ equal to 1% of the bounding box B of the scattered data $\mathbf{x} \in \mathbb{R}^s$, $s = 3$, i.e.

$$\delta = \frac{1}{100}\mathsf{B}, \qquad \mathsf{B} = \max_{s=1,2,3}\left[\max \mathbf{x}_s - \min \mathbf{x}_s\right].$$

Hence, the procedure of surface reconstruction via implicit interfaces works as follows:



(a) $\alpha = 2$ (b) $\alpha = 4$ (c) $\alpha = 6$

(d) $\alpha = 2$ (e) $\alpha = 4$ (f) $\alpha = 6$

Figure 4.5 | Demonstration of the influence of the weight function parameter α onto the quality of the reconstructed surface with $N = 163$ (top row) and $N = 2663$ (bottom row) Halton reference points.

(i) read the point cloud data set $\mathscr{S} : \{\mathbf{x}_i, \quad i = 1 \ldots N\}$,

(ii) read (or compute) the normals $\mathbf{n}_i, \quad i = 1 \ldots N$,

(iii) "extend" the data set, i.e. compute $\tilde{\mathscr{S}} : \{\tilde{\mathbf{x}}_i = \mathbf{x}_i \pm \delta \mathbf{n}_i, \quad i = 1 \ldots N\}$,

(iv) compute the interpolating function for the whole data set $\mathscr{S} \cup \tilde{\mathscr{S}}$,

(v) extract the zero-isosurface of the interpolant.

For a 3d point cloud the interpolant is a 4d hypersurface, and the partitional steps of the surface reconstruction are difficult to visualize. Hence, for the sake of simplicity we firstly demonstrate this approach on a 2d analogous problem and proceed to an example in 3d thereafter. Some of the results of the following Section are reported by the author in [154].

4.3.1 Demonstration of the approach in 2d

Given a point data set of a closed parametric curve $\mathscr{C}$ in a x-y-plane, we search for an interpolating curve $\hat{\mathscr{C}}$.

As an example we take the parametric curve

$$\mathscr{C}: \begin{cases} x(t) = r(t)\cos t, \\ y(t) = r(t)\sin t, \end{cases} \tag{4.4}$$

with a time-dependent radius $r(t) = 1 + \frac{1}{\gamma}\sin\gamma t$, and $t = 0 \ldots 2\pi$, with its outer normal $\mathbf{n} = (n_x, n_y)^\mathsf{T}$ easily found as

$$\begin{aligned} n_x(t) &= \dot{y}(t) = \cos\gamma t \sin t + \cos t \left(1 + \frac{\sin\gamma t}{\gamma}\right), \\ n_y(t) &= -\dot{x}(t) = -\cos t \cos\gamma t + \sin t \left(1 + \frac{\sin\gamma t}{\gamma}\right). \end{aligned} \tag{4.5}$$

Figure 4.6 shows the curve (4.4) for different parameters γ.

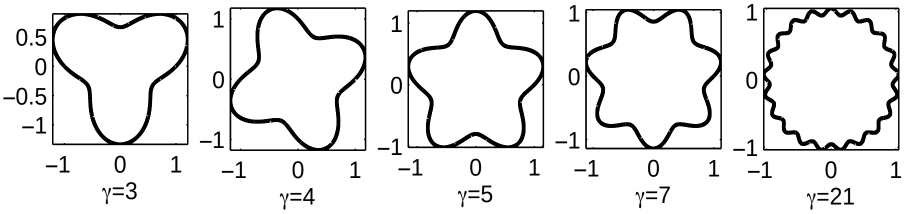
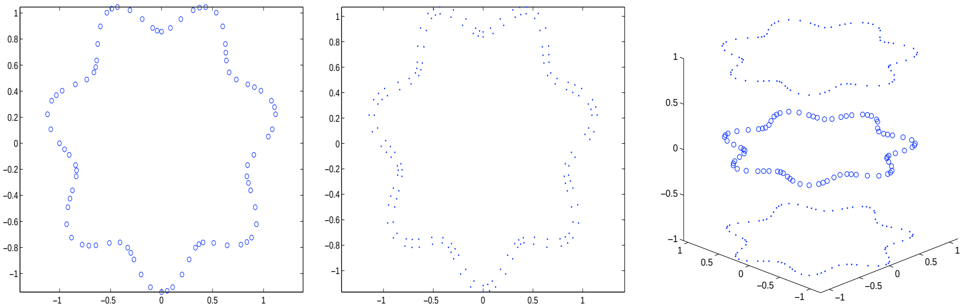


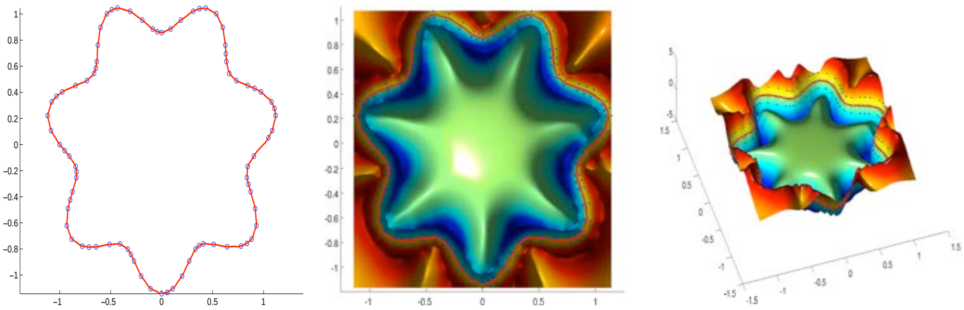
Figure 4.6 | A parametric curve for demonstration of interpolation of point cloud data in 2d via implicit surfaces for different parameters γ.

A point cloud data set is generated using the uniformly, randomly distributed N Halton points in 2d, so that the curve parameter $t = 2\pi \cdot \mathsf{H}_{2,N}$ (see Definition 3.3 on page 40). Using the normals (4.5) and the procedure described above, we compute the additional data points according to (4.1). Figure 4.7 on the facing page shows the interpolation of the 2d point cloud data for the case $\gamma = 7$. The original point cloud data, which consists of $N = 81$ Halton points distributed along the parametric curve (4.4) (see Figure 4.7(a)), is "extended", i.e. additional points are added along the normals (4.5) both inside and outside the curve (see Figure 4.7(b)).

Assigning values to the extended data ($\tilde{\phi} = \pm 1$) within and outside the curve $\mathscr{C}$, we interpolate it by reconstructing a higher dimensional (3d) surface and evaluating it on a regular 80×80 grid in $[-1, 1]^2$. The zero-isoline of this surface describes the sought interpolating curve $\hat{\mathscr{C}}$ (see the bottom row of Figure 4.7 on the next page) and Figure 4.8 on page 56



(a) *point cloud data* (b) *"extended" data* (c) *"extended" data values*



(d) *isoline* (e) *surface (top view)* (f) *surface (side view)*

Figure 4.7 | Interpolation of 2d point cloud data by an implicit surface.

demonstrates the influence of parameter δ onto the interpolation quality for the curve (4.4) with $\gamma = 3$. For this simple example going farther than 50% of the bounding box of the original data cloud data while extending the data set leads to loss of accurate resolution of the zero-isosurface. For more complicated surfaces in 3d this threshold value will, of course, be much smaller, and therefore the choice of δ is important.

4.3.2 Examples in 3d

Let us now look at the interpolation of point cloud data in 3d and consider the following examples. The first one is a standard testing surface for algorithms dealing with approximation of scattered data, and the second one is not only a nice-looking surface with analytical expression, but the one of importance both in nature and engineering applications.

► Error estimates and convergence analysis of approximations by IMLS are the subject of the following chapters and, therefore, here we restrict the investigations to a qualitative analysis of the reconstructed surfaces only, which is sufficient for defining a

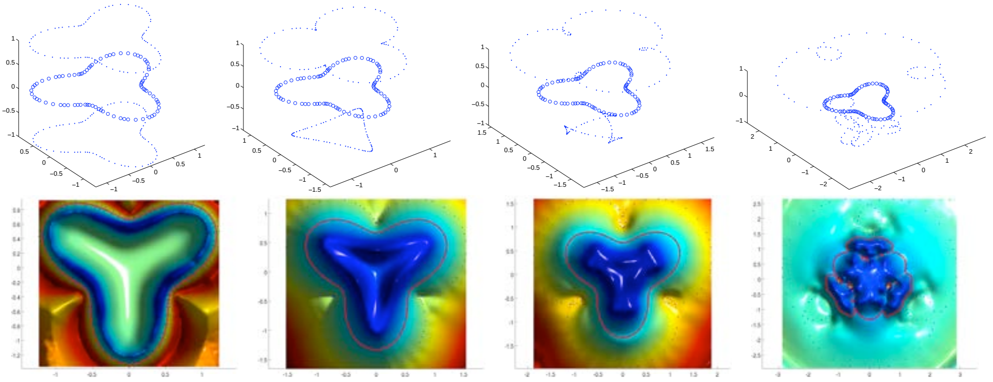


Figure 4.8 | Influence of the parameter δ onto the interpolation of 2d point cloud data by an implicit surface. Extended data sets (top row), and the corresponding interpolating surfaces (bottom row) from left to right for $\delta = 1\%, 10\%, 20\%, 50\%$ of the bounding box of point cloud data.

'mapping' procedure of distribution of computational points in a domain of interest with a complicated shape as described at the beginning of this chapter.

Example 1: Stanford bunny

The data sets used in this example correspond to a series of resolutions of the digitized surface of the Stanford bunny [2] – a test model from the field of computer graphics consisting of a digital image known also as a range scan (see Figure 4.9).

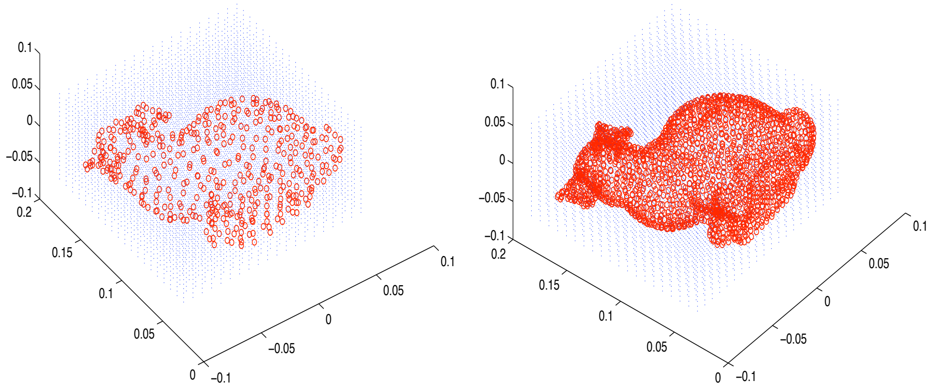
These sets of different resolutions consist of 453, 1889, 8171, and 35947 data points, respectively, and were taken together with the computed normals to the surface from [59], see Figure 4.10 on the facing page.

The implementation of this 3d example of surface reconstruction is very similar to the 2d example discussed in the previous section, only for visualization of results we are not able to represent the whole 4d hypersurface but rather the sought 3d zero-isosurface is demonstrated.

(a) *original object* (b) *digitized object surface*

Figure 4.9 | Stanford bunny.

The approximated values of the interpolating 4d hypersurface are evaluated on a set of n^3 points regularly distributed in the bounding box of the 3d point cloud data. The original data sets are extended analogously to the previous 2d example with parameter δ equal to 1% of the bounding box of the data. The overall data is then interpolated, and the extracted zero-isosurface represents the surface of the object.



(a) *453 data points* (b) *1889 data points*

Figure 4.10 | Stanford bunny: 3d reference data points (red circles) and the evaluation regular grid (blue dots) of $n^3 = 20^3$ evaluation points.

Figure 4.11 on the next page represents the reconstructed surfaces of the Stanford bunny evaluated for different mesh resolutions of n^3 evaluation points in the bounding box of the 3d point cloud data. For the set of 453 data points we used 8000, 15625, and 125000 evaluation points (left column), and for the set of 1889 data points we used 8000, 15625, and 27000 evaluation points (right column).

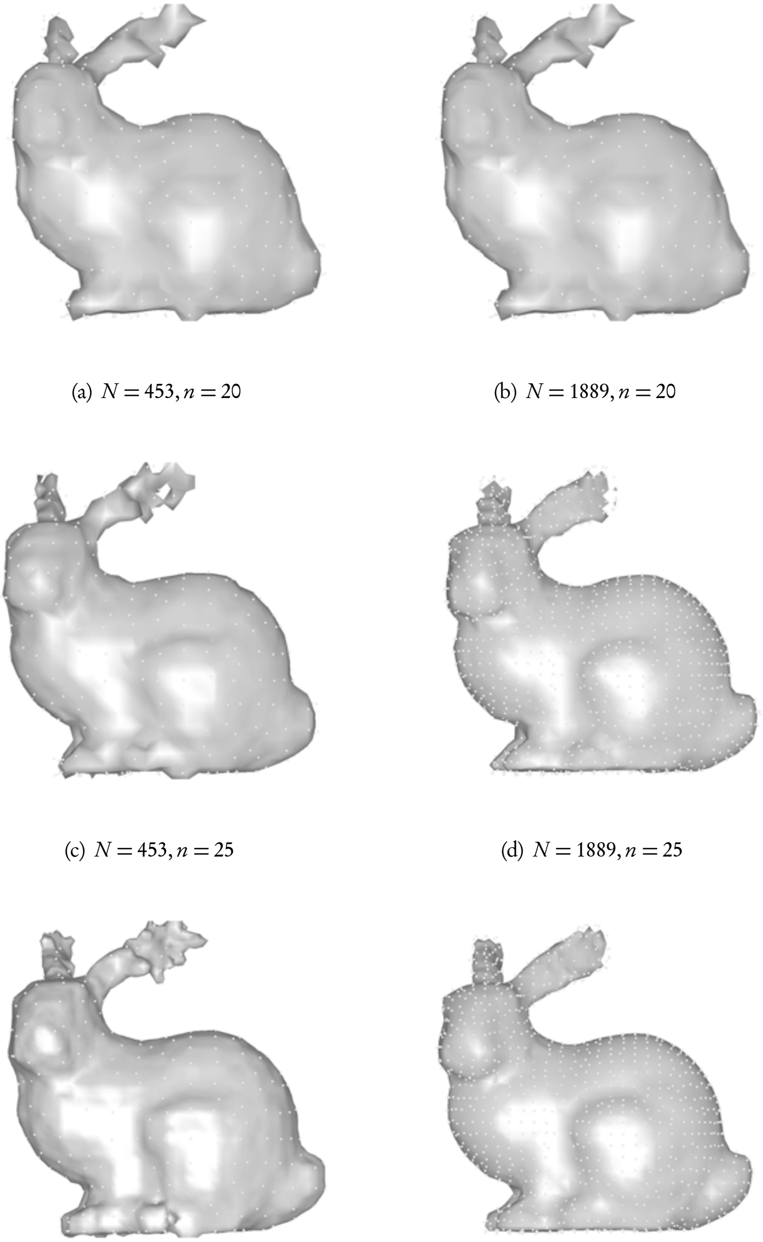
Figure 4.12 on page 59 shows detailed views on the reconstructed surface of the Stanford bunny with the finest available resolution of $N = 35947$ data points evaluated on a regular grid of $50^3 = 125000$ evaluation points. The object details are resolved very well, the holes at the bottom of the object (and consequently the holes in the data) can be observed in Figure 4.12(d), but due to the implicit technology the iso-surface is closed.

Example 2: Helix and double helix

Figure 4.13(a) shows a spiral staircase[1] at the entrance to Louvre Museum in Paris, France. Strictly speaking a spiral case is a helicoid (Figure 4.13(b)) – a surface obtained by letting a line rotate around an axis while simultaneously moving along it. The name 'helicoid' derives from its similarity to 'helix' (a surface shown in Figure 4.13(c)): for every point on the helicoid there is a helix contained in the helicoid which passes through that point.

---

[1] Photo by Toni Burkhardt.



(a) $N = 453, n = 20$

(b) $N = 1889, n = 20$

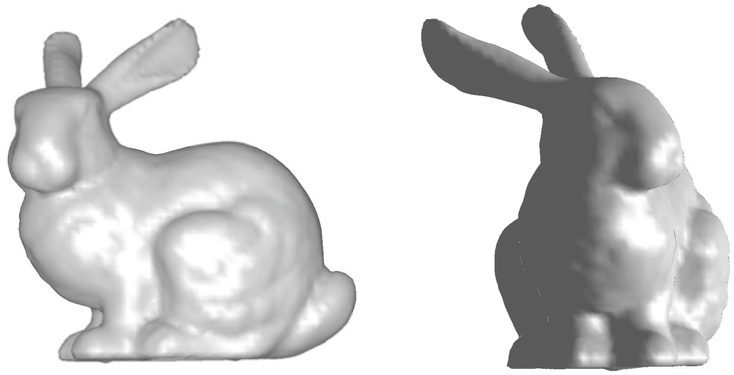
(c) $N = 453, n = 25$

(d) $N = 1889, n = 25$

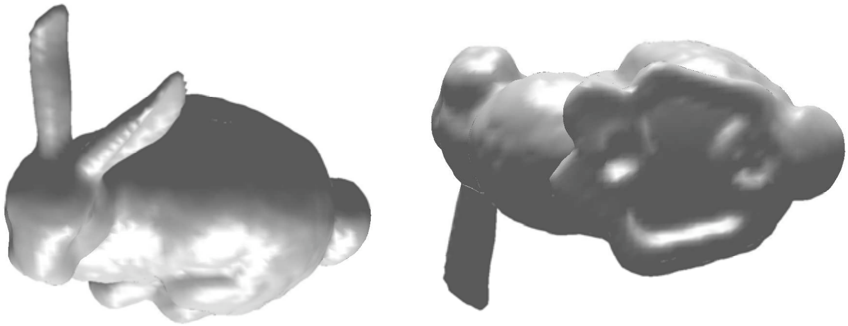
(e) $N = 453, n = 50$

(f) $N = 1889, n = 30$

Figure 4.11 | Reconstruction of a 3d point cloud data of N data points (white circles) of the digitized Stanford bunny surface by a zero-isosurface of an implicit 4d hypersurface evaluated on a regular grid of n^3 evaluation points.



(a) *side view* (b) *front view*



(c) *top view* (d) *bottom view*

Figure 4.12 | A detailed view on the reconstruction of the Stanford bunny with the finest available resolution of $N = 35947$ data points. An implicit 4d hypersurface was evaluated on a regular grid of $n^3 = 50^3 = 125000$ points.

There are many examples of helices in nature, engineering, and everyday life, e.g. climber plants, metal spiral, a slinky toy, corkscrew, screws and bolts, roller coaster, water slides, shells, deoxyribonucleic acid (DNA), Archimedes' screw (a machine originally utilized to transfer low-level water), novel wind turbines, etc.

The surface of a helix shown in Figure 4.13(c) is reconstructed by Interpolating Moving Least Squares, and the results for different resolutions of both available data and the evaluation grid are presented in Figure 4.14 on page 61. The reconstruction results of the surface of a double helix (a combination of two single helices) from a set of $N = 2 \times 225$ data points is given in Figure 4.15 on page 61. Analytical expressions of used helix surfaces and their normals are given in Appendix A on page 151. The data set consists of N Halton points, and in order to extend the data we used the analytical expressions for normals, and the re-

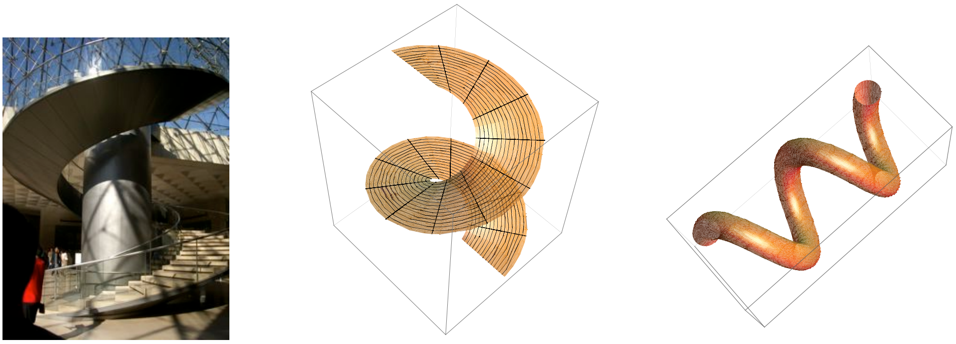

(a) *a spiral staircase* (b) *helicoid* (c) *helix*

Figure 4.13 | Helicoid and helix surfaces.

constructed surface is plotted as a zero-icocontour of the 4d interpolant.

Qualitatively we note that the extracted helix surface is choppy for a coarse evaluation grid, and approaches the exact solution as the number of evaluation points increases (see Figure 4.15 on the facing page). Yet the surface is rough due to a moderate number of reference data points ($N = 450$). If the number of data points is increased, the reconstructed surface is getting smoother. On open boundaries ('open ends' of the helices) the surface has special boundary points. At such points the proposed method tends to produce additional zero-levels, as can be seen from the reconstructed helix surfaces, e.g. Figure 4.15(c).

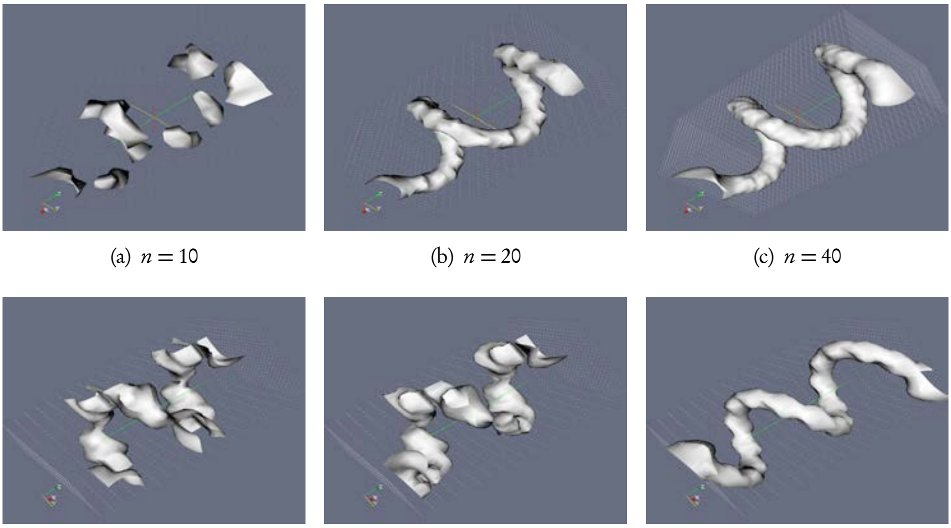
Quantitative studies on accuracy and convergence properties of the method are the subject of Chapter 5. If multiresolution subsets needed for this purpose are not available, they can be constructed from an original 3d point clouds with finest resolution via techniques found in [59]. Alternatively, a sophisticated and efficient algorithm for thinning 3d point clouds based on MLS proposed recently by Dyn et al. [49] can be utilized. In this approach points are recursively removed according to certain criteria based upon meshfree local surface reconstruction by RBFs and the original data is 'thinned'.

If the information about normals is not available, it must be computed from the point cloud data itself before the technique presented in this chapter can be applied. For an overview on techniques for this purpose see e.g. [203].

A simple object description format specially created for researchers working with polygonal models – the `.ply` format – is often used in computer graphics applications. For this format a comfortable utility for computing the normals – `normalsply` – by Greg Turk is available as a part of `ply-tools`[2].

---

[2] `http://www.cc.gatech.edu/projects/large_models/ply.html`


(a) $n = 10$ (b) $n = 20$ (c) $n = 40$

(d) $N = 49$ (e) $N = 90$ (f) $N = 7225$

Figure 4.14 | Interpolation of a 3d point cloud data of a helix. *Top row*: the number of uniformly, randomly distributed Halton data points $N = 225$, interpolating surface computed on a regular n^3 grid. *Bottom row*: the interpolating surface computed on a regular 20^3 grid for various number of Halton data points N.

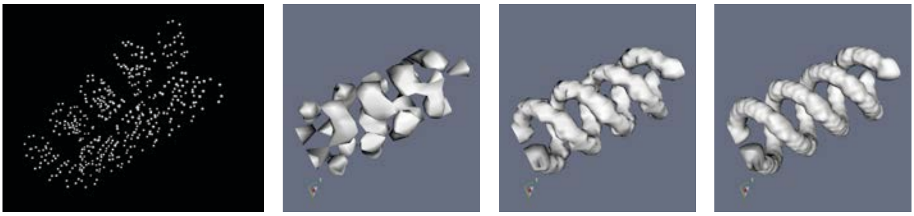

(a) *data set* $N = 2 \times 225$ (b) $n = 10$ (c) $n = 20$ (d) $n = 50$

Figure 4.15 | Interpolation of a 3d point cloud data of a double helix (a) evaluated from $N = 450$ Halton data points on a regular n^3 grid, i.e. 1000, 8000, and 125000 evaluation points.

"Learning is finding out what you already know."

-- Richard Bach

Chapter 5

Interpolating Functions and Derivatives

Before we apply the Space-Time Meshfree Collocation approach described in Chapter 3 to the solution of ordinary and partial differential equations in Chapter 6, let us investigate the accuracy of the method when applied to approximation of functions and their derivatives, since the latter is crucial for the accuracy of the solution of PDEs by collocations. Later on it will become clear that solving PDEs by collocations is a reciprocal procedure to scattered data interpolation (see Chapter 6 on page 73).

In all examples of this chapter we proceed in the following way: given an s-dimensional reference data points set $\mathscr{S} : \{\mathbf{x}_i \in \mathbb{R}^s,\ i = 1,\ldots,N\}$ and corresponding data values at these points $z_i \in \mathbb{R}$, the values of an interpolating function $z(\mathbf{x})$ and its derivatives $\mathscr{D}^{\alpha}_{\mathbf{x}} z(\mathbf{x})$ at an arbitrary point $\mathbf{x}_\ell \in \mathbb{R}^s$ are to be determined.

The main differences to the previous surface reconstruction examples are:

- not only the function values itself, but also its derivatives are to be computed,
- as data values at reference points we choose values of a function with a known analytical expression.

The latter makes it possible to compute the errors between the exact and approximated values. Of course, a computation of approximated values at *every* arbitrary point $\mathbf{x}_\ell \in \mathbb{R}^s$ is not possible. Therefore, we choose a subset of discrete evaluation points $\mathscr{E} \subset \mathbb{R}^s$, whereas the evaluation points in $\mathscr{E}$ are, in general, distinct from (but may also coincide with) the reference data points $\mathscr{S}$.

5.1 Error indicators and convergence

To measure the accuracy of our numerical method we will use several indicators commonly used in the literature to measure the errors between the numerical and exact values evaluated at a set of N discrete points, i.e. $\mathbf{u}^{\mathsf{appx}} = (u^{\mathsf{appx}}(\mathbf{x}_1),\ldots,u^{\mathsf{appx}}(\mathbf{x}_N))^{\mathsf{T}}$ and $\mathbf{u}^{\mathsf{ex}} = (u^{\mathsf{ex}}(\mathbf{x}_1),\ldots,u^{\mathsf{ex}}(\mathbf{x}_N))^{\mathsf{T}}$:

- the averaged maximum absolute error

$$\mathsf{E}_{\mathsf{abs}} := \frac{1}{N}\,||\mathbf{u}^{\mathsf{appx}} - \mathbf{u}^{\mathsf{ex}}||_{\infty}\,, \tag{5.1}$$

- the averaged maximum relative error

$$\mathsf{E}_{\mathsf{rel}} := \frac{1}{N}\,\frac{||\mathbf{u}^{\mathsf{appx}} - \mathbf{u}^{\mathsf{ex}}||_{\infty}}{||\mathbf{u}^{\mathsf{ex}}||_{\infty}}, \tag{5.2}$$

- the root-mean-square error (RMS-error)

$$\begin{aligned}\mathsf{E}_{\mathsf{rms}} &:= \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[u^{\mathsf{appx}}(\mathbf{x}_i) - u^{\mathsf{ex}}(\mathbf{x}_i)\right]^2} \\ &= \frac{1}{\sqrt{N}}\,||\mathbf{u}^{\mathsf{appx}} - \mathbf{u}^{\mathsf{ex}}||_2\,.\end{aligned} \tag{5.3}$$

► We measure the errors for a function and its derivatives in slightly different ways. The reason for this is the following. Since the method is interpolating, the numerical values of a test function *at reference data points* are exact, and a certain error is measured in between, i.e. *at evaluation points* only. For function derivatives, however, we derived in previous sections formulae *at reference data points* only. And in fact, we are interested only in these errors, since they are of importance for the collocation solution of PDEs. In this case the collocation points where the PDE is satisfied exactly coincide with the reference data points (see Section 6 on page 73). The values of derivatives can also easily be computed *at evaluation points* by a simple multiplication of values of the derivatives at reference points with the appropriate kernel IMLS functions of evaluation points, i.e. by interpolating the values of derivatives at reference points.

In order to test the convergence of the method, a series of uniformly random distributed Halton data reference points are generated in 2d ($s = 2$) as

$$\mathscr{S} : \left\{(2^k+1)^s\right\} = \{9, 25, 81, 289, 1089, 4225, 16641, 66049, 263169, \ldots\}, \tag{5.4}$$

and in 3d ($s = 3$) as

$$\mathscr{S} : \left\{(2^k+1)^s\right\} = \{27, 125, 729, 4913, 35937, 274625, 2146689, \ldots\}, \tag{5.5}$$

with $k \in \mathbb{N}$; the corresponding fill distances (computed according to (3.5) in Section 3.5 on page 43) are given in Table 5.1 on the facing page.

Table 5.1 | Series of Halton points on a unit square $[0,1)^s$ in 2d and 3d and the corresponding fill distances.

	$s=2$			$s=3$	
k	N	$h_{\Omega,\mathscr{S}}$	k	N	$h_{\Omega,\mathscr{S}}$
1	9	4.358987e-01	1	27	4.911522e-01
2	25	2.666864e-01	2	125	2.820103e-01
3	81	1.271675e-01	3	729	1.597701e-01
4	289	7.464144e-02	4	4913	9.465233e-02
5	1089	3.939036e-02	5	35937	4.261199e-02
6	4225	2.194550e-02	6	274625	2.346661e-02
7	16641	1.034171e-02	7	2146689	1.141810e-02
8	66049	4.493722e-03			
9	263169	2.459781e-03			

As a subset of discrete evaluation points $\mathscr{E} \subset \mathbb{R}^s$ we choose a regular grid of fine resolution. The constructed IMLS kernel functions, computed at these evaluation points, are used to approximate the values of a function under consideration.

The convergence behaviour of the solution is studied using the *experimental order of convergence* (EOC) defined by

$$\mathsf{EOC}^k := \frac{\ln\left[\mathsf{E}^{k-1} \,/\, \mathsf{E}^k\right]}{\ln\left[h^{k-1}_{\Omega,\mathscr{S}} \,/\, h^{k}_{\Omega,\mathscr{S}}\right]}, \quad k = 2, 3, \ldots, \tag{5.6}$$

where E^k is the appropriate error, e.g. the averaged maximum absolute, relative, or root-mean-square error as defined above in (5.1)–(5.3), for experiment number k, and $h^k_{\Omega,\mathscr{S}}$ the fill distance of the k-th data set.

▶ The goal of this chapter is to study the accuracy and convergence properties of STMCM for interpolation of a function with known analytical expression and to approximate its derivatives. The quality of approximation of the derivatives is decisive for the quality of the solution of differential equations by collocations using the STMCM.

Figures 5.1 on the next page and 5.4 on page 70 show the following test functions, their first and second derivatives w.r.t. x and y, and the Laplace operator, all evaluated according to analytical formulae from Appendix A on page 151. Here and in further examples the figure title Z denotes the function itself, and subindices denote appropriate partial derivatives.

5.2 Highly-oscillating functions :: Franke's function on a quadratic domain

As a test function for this example we choose a standard function used for 2d scattered data interpolation – Franke's function – and compute the first and second derivatives w.r.t. space coordinates x and y on a series of reference Halton data points (see Appendix A on page 151 for analytical expressions).

It is worthwhile noting that whereas the values of Franke's function and its first derivatives on a unit square $[0,1]^2$ vary roughly from -2 to 2, the values of the second derivatives and of the Laplace operator are one order magnitude higher, see Figure 5.1. This fact makes the Franke's function quite challenging when trying to approximate its values (and especially the values of the derivatives) from scattered data.

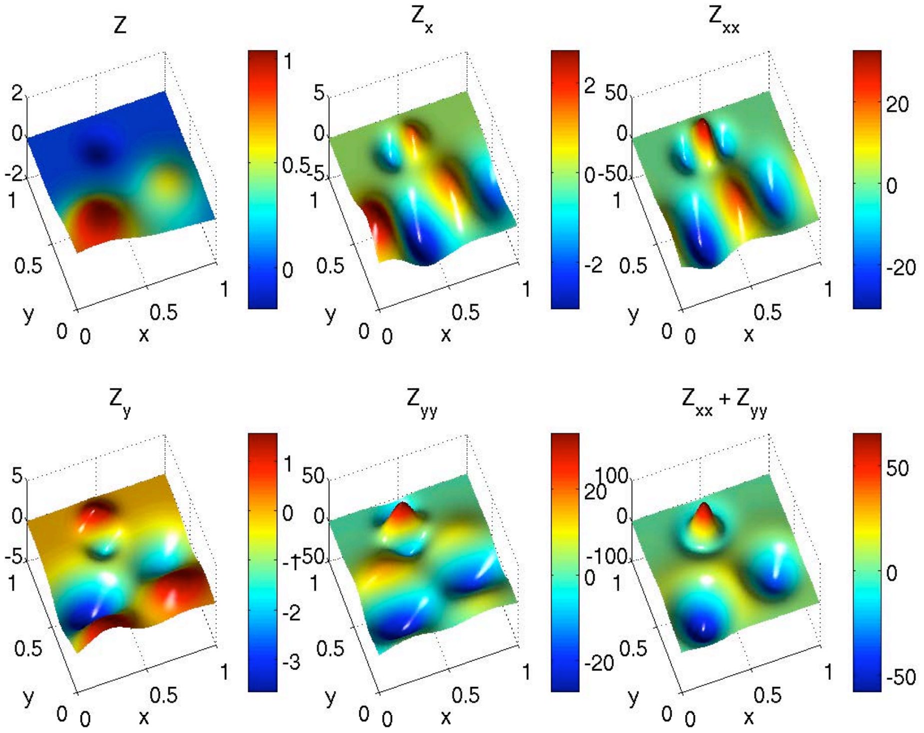


Figure 5.1 | Franke's test function and its derivatives.

Figure 5.2 on the facing page shows a series of reference data points generated from Halton sequences used to study convergence of the method when applied to interpolation of

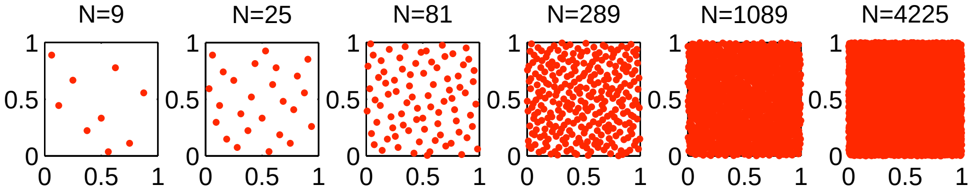


Figure 5.2 | A series of N Halton data point sets on a unit square for convergence studies.

Table 5.2 | Convergence study of Franke's function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
1	9	3.371869e+00	2.729173e-01	2.716908e-01	–	–
2	25	3.930370e-01	3.276987e-02	3.262260e-02	4.3744	4.3141
3	81	8.554396e-02	8.170292e-03	8.133574e-03	2.0591	1.8756
4	289	7.047104e-03	6.427983e-04	6.399095e-04	4.6854	4.7717
5	1089	1.694425e-03	1.619399e-04	1.612121e-04	2.2299	2.1569
6	4225	4.031413e-04	3.868432e-05	3.851046e-05	2.4546	2.4477

Franke's function and to the approximation of its first and second derivatives. At these sets of Halton points the function is interpolated, therefore the error is measured at a regular 80×80 evaluation grid. For derivatives, however, certain errors are measured both at the evaluation and reference sets of points. The values for derivatives are obtained by a product of the appropriate derivatives of the kernel functions and the values of the function at reference points.

The results of the convergence study of Franke's function and its derivatives are demonstrated in Tables 5.2-5.4 and the convergence plots of errors vs. the fill distance of the reference data are shown in Figure 5.3 on page 69. For the RMS-error as well as for the averaged absolute error of Franke's function we obtained roughly the expected third order of convergence. The experimental orders of convergence for the first and second derivatives are, as expected, of second and first order, respectively.

- ▶ In this chapter we used the following IMLS parameters: $\alpha = 8$, $\rho = 2.5h_{\Omega,\mathscr{S}}$, and quadratic basis $q = 2$ (see Section 3.5.1 on page 43 for details). The influence of parameters onto the quality of solution will be studied in the next sections.

Figure 5.3 on page 69 demonstrates the convergence behaviour of approximation of the Franke's function and its derivatives.

Table 5.3 | Convergence study of the first derivatives of Franke's function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
			w.r.t. x			
1	9	1.132906e+00	3.132958e-01	1.049036e-01	–	–
2	25	9.093525e-01	1.075269e-01	3.600419e-02	0.4474	2.1765
3	81	2.765849e-01	9.192810e-03	3.078111e-03	1.6072	3.3209
4	289	9.062182e-02	1.429441e-03	4.786324e-04	2.0942	3.4931
5	1089	2.600579e-02	1.456365e-04	4.854666e-05	1.9531	3.5732
6	4225	7.299412e-03	1.302719e-05	4.301816e-06	2.1720	4.1269
			w.r.t. y			
1	9	9.946226e-01	2.419503e-01	9.465171e-02	–	–
2	25	7.185588e-01	8.688638e-02	2.784422e-02	0.6617	2.0844
3	81	2.526078e-01	9.654885e-03	2.674979e-03	1.4116	2.9668
4	289	9.004119e-02	1.474727e-03	4.085873e-04	1.9361	3.5266
5	1089	2.550806e-02	1.712841e-04	4.722514e-05	1.9733	3.3683
6	4225	6.538722e-03	1.364028e-05	3.744847e-06	2.3271	4.3256

Table 5.4 | Convergence study of the second derivatives of Franke's function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
			w.r.t. x			
1	9	8.351651e+00	1.769577e+00	9.019159e-02	–	–
2	25	9.199613e+00	9.571041e-01	3.331590e-02	0.1968	1.2508
3	81	4.597547e+00	2.229804e-01	7.510931e-03	0.9366	1.9672
4	289	2.186708e+00	3.450886e-02	1.129702e-03	1.3947	3.5019
5	1089	8.074090e-01	5.461888e-03	1.683369e-04	1.5588	2.8841
6	4225	3.035810e-01	6.810342e-04	2.098967e-05	1.6722	3.5591
			w.r.t. y			
1	9	8.700501e+00	1.728473e+00	1.044060e-01	–	–
2	25	7.484086e+00	6.591047e-01	2.886886e-02	0.3065	1.9622
3	81	4.069652e+00	2.107938e-01	8.298009e-03	0.8226	1.5394
4	289	1.873128e+00	3.724217e-02	1.219004e-03	1.4563	3.2534
5	1089	6.183491e-01	3.423469e-03	1.044964e-04	1.7340	3.7342
6	4225	2.947582e-01	5.736437e-04	1.750964e-05	1.2666	3.0539

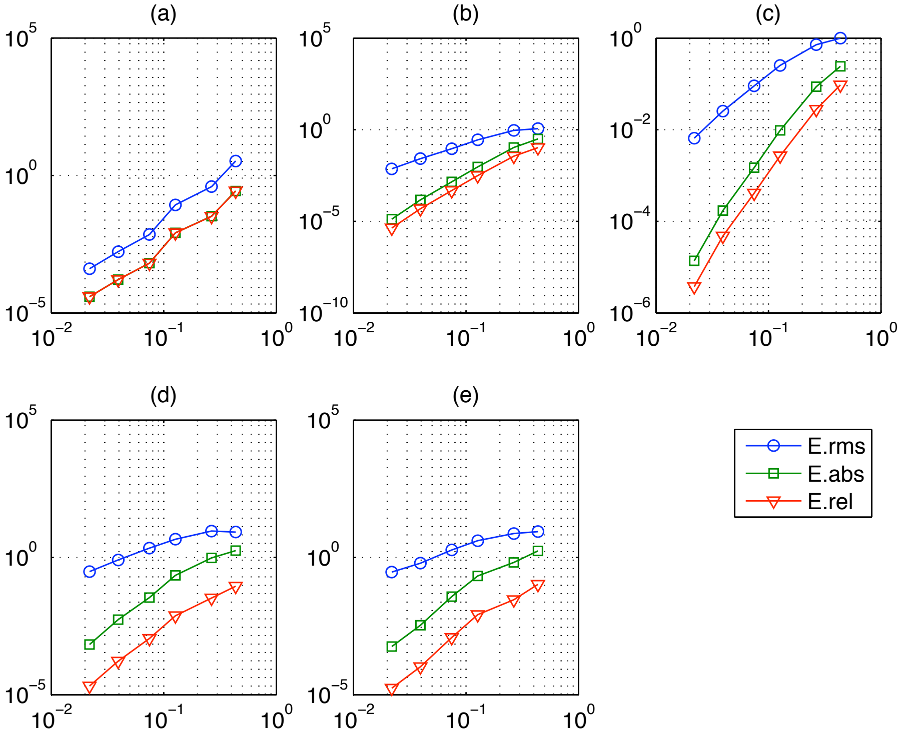


Figure 5.3 | Convergence plots (errors vs. fill distance) of Franke’s function (a), its first derivatives w.r.t x (b) and y (c), and second derivatives w.r.t x (d) and y (e), respectively.

5.3 Highly-oscillating functions :: A “blupp” function on a circular domain

The test function for this example is the so called “blupp” function defined in a circular domain of radius 8, where it exhibits steep gradients (see Figure 5.4 on the following page).

Also in this example we performed a convergence analysis on a series of Halton data points (see Appendix A on page 151 for analytical expressions). In contrast to the previous example, the points are distributed in a circular domain of radius 8 as shown in Figure 5.5 and the corresponding fill distances, computed according to the definition from Section 3.5 on page 43, are given in Table 5.5 on the next page.

The results of the convergence study of the “blupp” function and its test derivatives are shown in Tables 5.6-5.8. Similarly to the results for Franke’s function we obtain in this case the third order of convergence for the function, and the second and first order for the

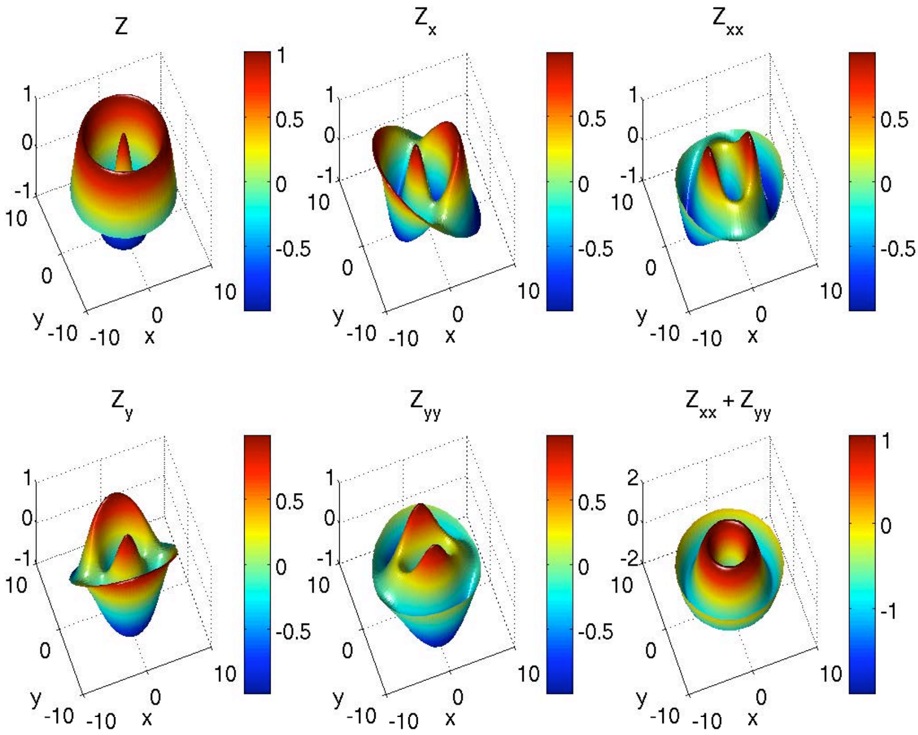


Figure 5.4 | A “blupp” test function and its derivatives.

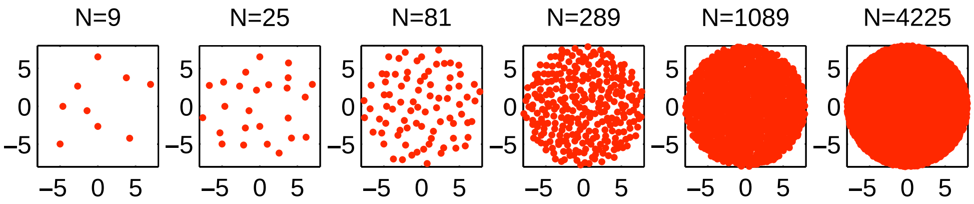


Figure 5.5 | A series of Halton data points set in a circular domain of radius 8 for convergence studies.

Table 5.5 | Fill distances for Halton points distributed in a circular domain of radius 8.

N	9	25	81	289	1089	4225
$h_{\Omega,\mathscr{S}}$	5.1557	3.2482	1.6457	1.0610	0.5755	0.2393

RMS-error of its first and second derivatives, respectively. Figure 5.6 on the following page demonstrates the convergence behaviour of approximation of the “blupp” function and its derivatives.

Table 5.6 | Convergence study of the “blupp” function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
1	9	9.104409e-01	2.056049e-02	2.056098e-02	–	–
2	25	6.926077e-01	2.536289e-02	2.536350e-02	0.5338	-0.4097
3	81	1.203614e-01	3.667741e-03	3.667829e-03	2.4510	2.7084
4	289	2.358009e-02	1.088418e-03	1.088444e-03	2.2770	1.6970
5	1089	3.651762e-03	1.243489e-04	1.243519e-04	4.0025	4.6553
6	4225	6.214383e-04	2.377203e-05	2.377260e-05	3.1702	2.9620

Table 5.7 | Convergence study of the first derivative of the “blupp” function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
			w.r.t. x			
1	9	7.115274e-01	1.134530e-01	1.139516e-01	–	–
2	25	4.276879e-01	3.896291e-02	3.913416e-02	0.9936	2.0862
3	81	1.803668e-01	7.102190e-03	7.133406e-03	1.2093	2.3841
4	289	6.337735e-02	6.477806e-04	6.502536e-04	1.4610	3.3450
5	1089	2.193219e-02	7.425259e-05	7.425616e-05	2.2772	4.6482
6	4225	6.223893e-03	7.027142e-06	7.027480e-06	2.2548	4.2206
			w.r.t. y			
1	9	4.582919e-01	8.468959e-02	1.472579e-01	–	–
2	25	4.561540e-01	3.619002e-02	3.965492e-02	0.9127	1.6595
3	81	1.625428e-01	5.601454e-03	5.711416e-03	1.4453	2.6132
4	289	6.470627e-02	6.783630e-04	6.785703e-04	1.2866	2.9490
5	1089	2.189787e-02	8.762532e-05	8.763117e-05	2.3250	4.3918
6	4225	6.178700e-03	6.362019e-06	6.362444e-06	2.2650	4.6951

▶ In this chapter the accuracy and convergence properties of the IMLS method for interpolation of functions and their derivatives was studied. The latter are used in the approach proposed in this thesis for a discretization of (partial or ordinary) differential equations in a strong form. Thus, the results of this chapter form a solid basis for numerical studies performed in the following chapters.

Table 5.8 | Convergence study of the second derivatives of the “blupp” function

k	N	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}
			w.r.t. x			
1	9	3.444497e-01	6.101127e-02	1.265293e-01	–	–
2	25	3.149248e-01	3.824921e-02	4.010649e-02	0.1749	0.9114
3	81	2.062448e-01	7.612413e-03	7.982052e-03	0.5928	2.2611
4	289	1.355837e-01	2.173850e-03	2.177707e-03	0.5860	1.7507
5	1089	7.799359e-02	4.011154e-04	4.018270e-04	1.1866	3.6266
6	4225	3.371569e-02	6.828979e-05	6.834550e-05	1.5013	3.1694
			w.r.t. y			
1	9	5.949938e-01	1.180248e-01	1.217740e-01	–	–
2	25	4.624316e-01	4.284623e-02	4.420729e-02	0.4919	1.9778
3	81	2.443490e-01	7.434265e-03	7.579596e-03	0.8935	2.4532
4	289	1.395879e-01	2.331546e-03	2.339065e-03	0.7821	1.6198
5	1089	7.686600e-02	4.421089e-04	4.432246e-04	1.2803	3.5681
6	4225	3.316662e-02	6.443176e-05	6.443320e-05	1.5047	3.4477

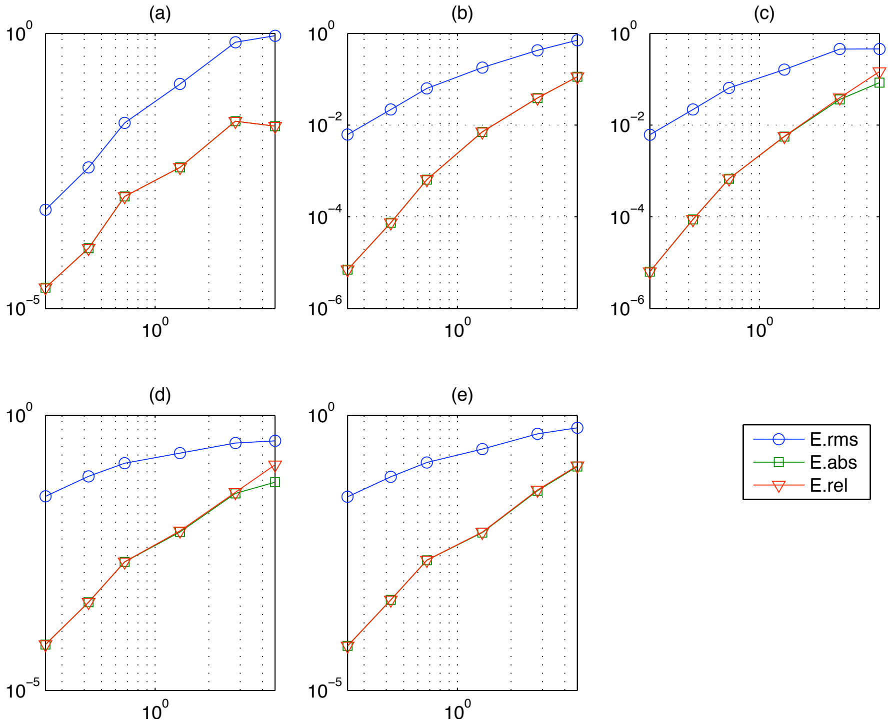


Figure 5.6 | Convergence plots (errors vs. fill distance) of the “blupp” function (a), its first derivatives w.r.t x (b) and y (c), and second derivatives w.r.t x (d) and y (e), respectively.

"Thus the partial differential equation entered theoretical physics as a handmaid, but has gradually become mistress."

-- Albert Einstein

"Someone told me that each equation I included in the book would halve the sales."

-- Stephen Hawking

CHAPTER 6

Numerical Study of STMCM for Differential Equations

We start this chapter by giving an overview of classes of differential equations found in mathematical modelling of diverse engineering problems. Then, we apply the STMCM presented in Chapter 3 to systems of linear and nonlinear ordinary differential and partial equations of various types and study the accuracy and convergence properties of the proposed method.

6.1 Introduction

In Chapter 4 we dealt with the surface reconstruction of scattered data and in Chapter 5 we investigated the quality of interpolation of functions and their derivatives. Numerical solution of PDEs by collocations can be seen as a reciprocal problem to that of interpolation of scattered data. In collocation methods we deal with a system of PDEs, which our solution has to satisfy at a set of nodes (reference points in terms of interpolation of scattered data). This system of equations contains the partial derivatives of the unknowns. Whereas in the interpolation problem we know the nodal values and look for values of the derivatives, the goal herein is to find the nodal values, while the values of the derivatives (or their combinations) are known.

The major part of all mathematical models dealing with different phenomena in engineering, physics, chemistry, biology, but also weather forecasting, image processing, etc. are described by PDEs, i.e. equations which contain partial derivatives of the unknowns w.r.t. space and/or time. The special case of PDEs are ordinary differential equations (ODEs) containing the time derivatives only.

In most cases these models are much too complicated to be solved analytically and the numerical simulations have become a standard tool to study the behaviour of different phenomena. Before we go into detail applying the proposed novel method and study its accuracy and convergence properties let us first briefly classify the underlying PDEs as different aspects have to be taken into account when dealing with equations of different types.

6.2 Classification of equations

PDEs are often classified into three groups due to different mathematical and physical behaviours: *elliptic* (diffusion and equilibrium problems), *parabolic* (time- or space-marching problems) and *hyperbolic* (propagation problems) equations. Introducing simplifications to the model, the type might change (e.g. considering the steady state of the flow a parabolic problem becomes an elliptic one, or considering a boundary layer phenomenon the type may change from an elliptic to a parabolic one).

6.2.1 Problem types

The major problems which occur during modelling of physical phenomena and engineering systems are:

(a) *Diffusion Problems*

Diffusion describes *irreversible processes* wherein the system moves towards the *most probable state*, e.g. high temperature ⇒ low temperature (thermal diffusion), or high concentration ⇒ low concentration.

If we look at a container with two fluids A and B sketched in Figure 6.1,

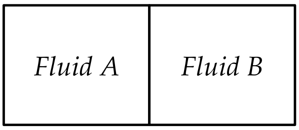


Figure 6.1 | Diffusion of two fluids.

we may proceed with the following considerations.

The probability that a particle of fluid A remains in A is $p = \frac{1}{2}$.

Then, if the overall number of particles is for instance $N = 200$, the probability that the fluids A and B *do not* mix is $p_{\text{no mix}} = \left(\frac{1}{2}\right)^N = \left(\frac{1}{2}\right)^{200} = 6.22\,\text{e}-61$, meaning that eventually they *will* mix.

(b) *Equilibrium Problems*

Equilibrium problems are usually boundary value problems (BVPs) and belong to the class of elliptic PDEs. They are usually steady-state problems (derivatives w.r.t. time are being neglected) with smooth gradients, whereas the entire solution must satisfy all boundary conditions (BCs).

The physical examples of equilibrium problems are the steady temperature distributions, steady viscous flow problems, and equilibrium in elastic structures.

(c) *Eigenvalue Problems*

Eigenvalue problems (EVPs) can be seen as an extension of equilibrium problems, wherein some critical values of certain parameters must be established.

These kinds of problems occur for example when studying buckling and stability issues of structures, vibration analysis or the effect of resonance in acoustics or in electric circuits.

(d) *Propagation Problems*

These initial value problems (IVPs) are unsteady, transient marching problems (meaning that the inverse problem is unstable) of a *hyperbolic* (solution is marching from an initial state and is being modified by the side BCs) or *parabolic* type (solution is marching in certain direction, and in other direction the equilibrium takes place).

Typical physical examples of propagation problems are the propagation of heat, of pressure waves in fluids, of displacements and stresses in elastic structures, and the development of self-induced vibrations.

6.2.2 Well-Posedness

While dealing with mathematical models and solving them numerically the important notion of *well-posedness* of the problem must be introduced. We distinguish between *mathematical* and *numerical well-posedness*.

Definition 6.1 *A system of governing equations is well-posed* mathematically *iff*

(i) a solution exists,

(ii) the solution is unique,

(iii) the solution depends continuously on the auxiliary data.

▶ Underdefined boundary and initial conditions lead to non-unique solutions, and overdefined to unphysical ones.

Definition 6.2 *A* discretized *system of governing equations and auxiliary data is well-posed* numerically *iff*

(i) an approximate solution exists,

(ii) the approximate solution is unique,

(iii) the approximate solution depends continuously on the discretized auxiliary data (boundary and initial conditions),

(iv) the numerical algorithm is stable.

The problems considered in the following sections are well-posed in the sense described above.

6.2.3 Equation types

In this section we would like to summarize the important properties of different types of equations according to the classification given above, since various aspects must be considered during the analysis of these three types and their numerical solutions.

I | **Hyperbolic equations**

- two propagation (marching) directions x and y,
- derivatives are not necessarily uniquely defined along the characteristics,
- discontinuities may occur.

II | **Elliptic equations**

- derivatives can always be uniquely defined for each point in the computational domain,
- no marching or propagation direction,
- smooth solutions.

III | **Parabolic equations**

- one characteristic direction (typically $t =$const),
- given the initial condition the solution is marching in time (or in space),
- the solution is modified by (time-dependent) BCs during the propagation,
- any change in BCs at $t = \hat{t}$ affects the solution for $t > \hat{t}$, i.e. *what already happened cannot be changed, but the future can be controlled.*

► Sometimes the problems may change their types. A typical example is an approximation of a boundary layer in fluid mechanics problems, wherein the overall problem is of an equilibrium type, while the boundary layer is approximated by parabolic equations.

In the following sections of this chapter we will apply the STMCM to various classes of ordinary and partial, linear and nonlinear differential equations and investigate its accuracy and convergence properties. The discussion of the convergence rates is provided at the end of the chapter in Section 6.8 on page 129.

6.3 Linear Ordinary Differential Equations

Behaviour of dynamical real-world systems that continuously evolve in time is described by ordinary differential equations.

6.3.1 A spring-mass oscillator

We start with a simple example – a second order ODE – the equation describing the motion of a spring-mass oscillator (see Figure 6.2 on the next page) with the mass m, the damping coefficient b, and the stiffness of the spring k:

$$m\ddot{u} + b\dot{u} + ku = f(t), \qquad \text{for} \quad t \in [0, T] \tag{6.1}$$

with the initial conditions for displacement u and velocity v

$$u|_{t=0} = u_0, \qquad \dot{u}|_{t=0} = v|_{t=0} = v_0. \tag{6.2}$$

An external periodic force $f(t) = \hat{f}\sin(\Omega t)$ is acting on the pendulum and sets it into motion.

- The same ODE describes e.g. a resonant electrical circuit consisting of inductor L and capacitor C, movement of a linearized pendulum and other harmonic oscillators.

An analytical solution of the system response can easily be found as

$$u(t) = e^{-\delta t}\left(\hat{a}_c \cos\omega t + \hat{a}_s \sin\omega t\right) + \hat{u}_c \cos\Omega t + \hat{u}_s \sin\Omega t, \tag{6.3}$$

with the damping factor $\delta := b/2m$ and natural frequency of the oscillator $\omega = \sqrt{k/m}$. Applying the initial conditions (6.2) we obtain the expressions for the unknown coefficients

$$\hat{u}_c = -\hat{f}\frac{\Omega b}{(k-\Omega^2 m)^2 + (\Omega b)^2}, \tag{6.4}$$

$$\hat{u}_s = \hat{f}\frac{(k-\Omega^2 m)}{(k-\Omega^2 m)^2 + (\Omega b)^2}, \tag{6.5}$$

and

$$\hat{a}_c = u_0 - \hat{u}_c, \tag{6.6}$$

$$\hat{a}_s = \frac{1}{\omega}(v_0 + \delta\hat{a}_c - \Omega\hat{u}_s). \tag{6.7}$$

Rewriting the second-order ODE (6.1) as a system of two first-order ODEs

$$m\dot{v} + bv + ku = f(t), \tag{6.8}$$

$$v - \dot{u} = 0, \tag{6.9}$$

and incorporating both the displacement u and the velocity v into a vector of unknowns, we can now apply the STMCM as presented in Chapter 3, whereas the method reduces to a meshfree collocation method in time, since only the time derivatives must be discretized. Hereby the whole time interval is discretized in one single step. In further sections dealing with linear and nonlinear coupled systems we will apply the time-slab stepping procedure and study the convergence and accuracy properties of both approaches (see Section 6.3.2 on page 82).

We consider the following cases and study the accuracy of the method comparing the result with the known analytical solution.

The parameters of the system are: $m = 300$, $k = 4800$, $b = 0$ (undamped case), $b = 600$ (damped case). The initial conditions used are: $u_0 = 0$, $v_0 = 2$.

We compute the solutions for two cases: a continuous (equation (6.10)) and a discontinuous (equation (6.11)) right-hand side of the equation of motion (6.8):

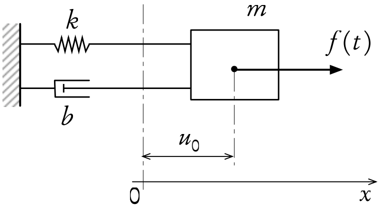


Figure 6.2 | A spring-mass oscillator.

$$f_1(t) = \begin{cases} 1800\sin(2t) & \text{for } t \geq \pi, \\ 0 & \text{else.} \end{cases} \tag{6.10}$$

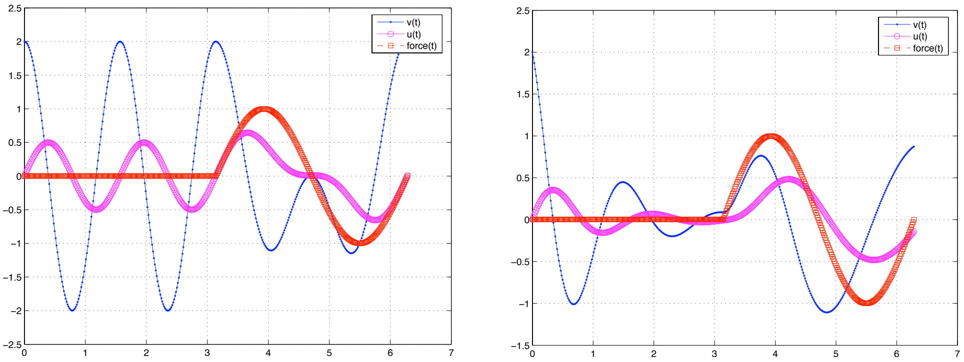
$$f_2(t) = \begin{cases} 1800 & \text{for } t \geq \pi, \\ 0 & \text{else.} \end{cases} \tag{6.11}$$

Figure 6.3 on the next page shows the solution of the equation of motion with the continuous right-hand side (6.10) for both undamped and a damped case. The computational solution is in excellent agreement with the analytical one (represented on the figure by a solid line). The chosen discretization parameters for this example were: 7 points in the support of the weight function (2.18), $\alpha = 4$, the time step was $\Delta t = 0.015$.

Figures 6.4 and 6.5 show the response of the system onto the excitation force with a jump (6.11) for the undamped and damped cases respectively. In both cases unphysical oscillations appear if a single time interval is used to simulate the jump. The problem can be overcome using two adjacent time intervals to simulate the jump.

The reason for these spurious oscillation is, however, the discontinuity in the external force, i.e. the right-hand-side of the equation of motion of the system. Since we solved the equations for the whole time interval from 0 to 2π *in one single step*, we observe that the "future" affects the "past" (the jump at later time instant produces artificial oscillations of the system response at earlier time instants).

The quality of discontinuity resolution can be increased if adaptive techniques are applied (the steeper the gradient, the more points are used). In addition, for more complicated systems one usually subdivides the whole time interval into time-slabs and applies a discretiza-



(a) *undamped* (b) *damped*

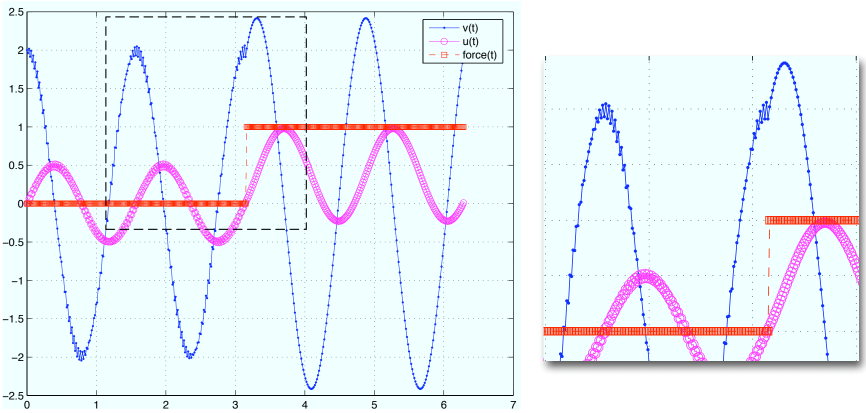
Figure 6.3 | ODE solution with a continuous right-hand side.

tion procedure (in our case as described in Section 3). We will compare these two possibilities and study the accuracy and convergence properties of the method in the following sections.

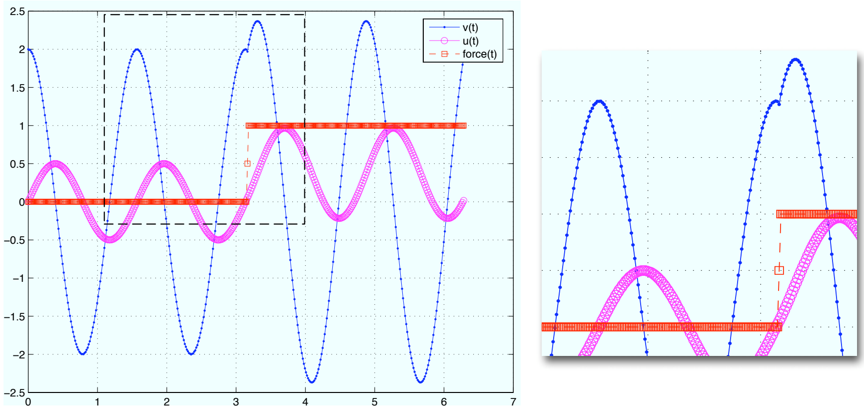
In order to study the quality of the numerical solution we perform here and in a number of the following examples a convergence study and compute the errors between the numerical and exact solutions according to guidelines given in Section 5.1 on page 63. We restrict this study to the undamped case with the right-hand side (6.10), the convergence behaviour for the other cases is of the same order. Table 6.1 shows the respected results (7 points in the support of the weight function, $\alpha = 8$, linear basis), we obtained the expected second order of convergence for the RMS- and the third order of convergence for the absolute and relative averaged errors, respectively. The influence of the IMLS parameters (i.e. α, ρ) onto the quality of the solution will be studied in the following sections.

Table 6.1 | Convergence results for the linear ODE (6.8) for a spring-mass oscillator.

Δt	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
0.5	3.2992e-01	4.1320e-02	8.3489e-02	–	–	–
0.25	3.4350e-01	2.0669e-02	4.7734e-02	-0.5822	0.9993	0.8066
0.1	1.6437e-01	6.5615e-03	1.3127e-02	0.8044	1.2523	1.4089
0.05	4.1635e-02	7.8597e-04	1.5740e-03	1.9811	3.0615	3.0601
0.025	9.8973e-03	9.3439e-05	1.8689e-04	2.0727	3.0724	3.0741
0.01	1.3821e-03	5.4449e-06	1.0890e-05	2.1485	3.1023	3.1024
0.005	3.4471e-04	6.7934e-07	1.3587e-06	2.0034	3.0027	3.0027

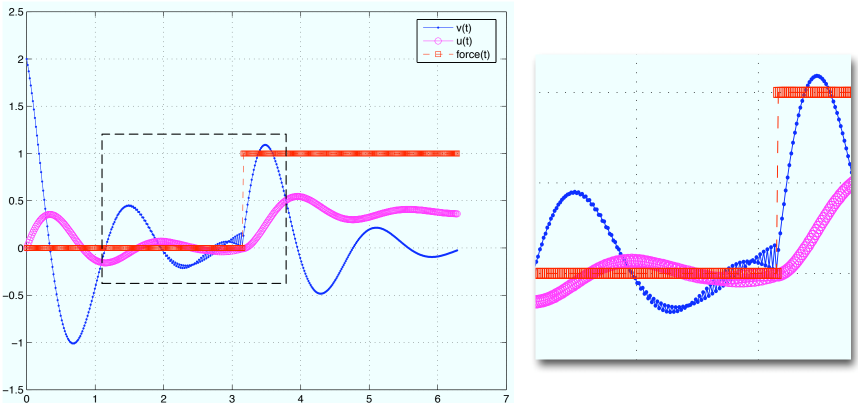


(a) *one interval to simulate the jump*

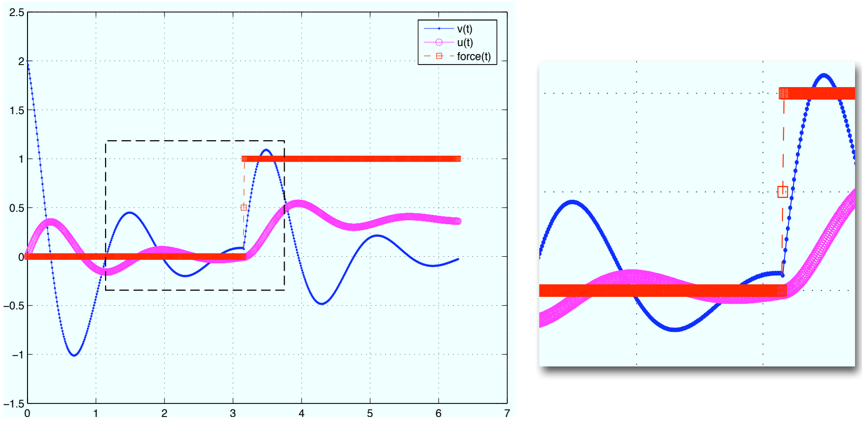


(b) *two intervals to simulate the jump*

Figure 6.4 | ODE undamped solution with a discontinuous right-hand side.



(a) *one interval to simulate the jump*



(b) *two intervals to simulate the jump*

Figure 6.5 | ODE damped solution with a discontinuous right-hand side.

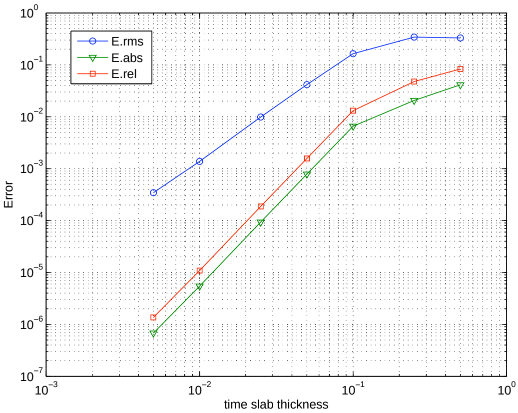


Figure 6.6 | Convergence study for the linear ODE (6.8) for a spring-mass oscillator.

6.3.2 Foucault pendulum

With the help of a simple system that consists of a rather long rod (67 m) and a bob attached to it the French physicist Jean Bernard Léon Foucault (1819–1868) observed that the plane of oscillations of the pendulum exhibited a rotation itself, which proved the rotation of the Earth. Figure 6.7 shows a smaller copy of the original Foucault Pendulum located in the church Saint Martin des Champs at Conservatoire Nacional des Arts et Métiers (CNAM) in Paris[1].

The period of rotation of the oscillation plane τ depends on the latitude of the pendulum γ as

$$\tau = \frac{24}{\sin\gamma}\ [\text{hrs}],$$

and for the latitude of Paris ($\gamma = 48.8°$) the pendulum makes a full cirle in $\tau = 31,89$ hours, i.e. the plane of oscillations rotates about $11,3°/\text{hr}$.

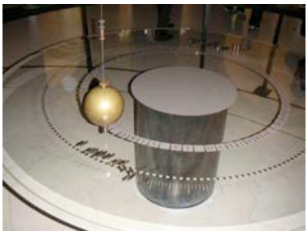


Figure 6.7 | Foucault Pendulum at Conservatoire Nacional des Arts et Métiers (CNAM), Paris.

The period of oscillation of the pendulum is given by

$$T = 2\pi\sqrt{\frac{L}{g}},$$

where L is the length of the cord in meters, and $g = 9.8\ \text{m/s}^2$ is the gravitational acceleration.

---

[1] The image was taken according to the GNU Free Documentation Licence (GFDL) from http://en.wikipedia.org/wiki/Foucault's Pendulum

Hence, there are two periodic movements in the system: one corresponding to an oscillation of a pendulum with the period T and the second one corresponding to the rotation of the plane of oscillation with the period τ. Interested readers are referred to [17] for historical remarks as well as the detailed derivation of equations of motion of the Foucault pendulum. Adapting the simplification assumptions therein (neglecting quadratic terms of the angular velocity of the Earth Ω, the centrifugal force, and taking into account the fact that the pendulum's length L is much larger than the displacements in z-direction), we end up with a simplified coupled linear system of equations of motion of the Foucault pendulum:

$$\begin{aligned} \dot{x} &= u, \\ \dot{u} &= -\omega^2 x + 2\sin\gamma\Omega\dot{y}, \\ \dot{y} &= v, \\ \dot{v} &= -\omega^2 y - 2\sin\gamma\Omega\dot{x}, \end{aligned} \tag{6.12}$$

where x and y are the displacements of the pendulum bob, u and v are its velocities in the respective direction, $\omega = 2\pi/T$ is the angular frequency of the pendulum, and Ω is the rotational frequency of the Earth. We use the parameters of the original Foucault pendulum in Paris, i.e. $L = 67$ m, $\gamma = 48.8°$. The rotation of the Earth is artificially "accelerated" in order to clearly demonstrate the effect, and we choose $\Omega = 0.02\ \mathrm{s}^{-1}$.

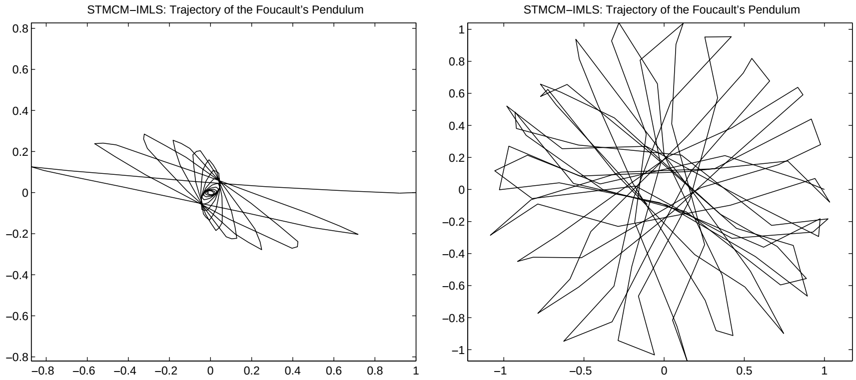
The STMCM reduces in the case of ODEs to a MCM in time. As a reference solution of the system (6.12) we choose a multivalue variable-order Adams-Bashforth-Moulton predictor-corrector (ABM-PECE) method, and solve the system on the interval $t \in [0, 200]$ with the initial conditions $x_0 = 1$ and $y_0 = u_0 = v_0 = 0$.

We shall now look at two different procedures. The first one resembles to a *onestep*-like time integration method, i.e. we discretize a single time slab with a minimum of points required by the basis. In this case the time-slab thickness must be rather small in order to capture the solution. The second possibility is to solve the systems of equations once for the whole time interval and hereby we are dealing with a *multistep*-like time integration method. In this case the number of points to discretize the whole time interval must be rather high.

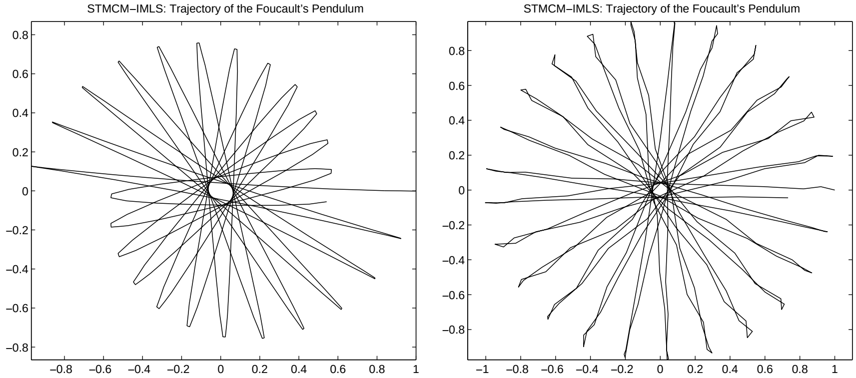
We demonstrate in the following that the method works successfully for both approaches. For different setups of parameters we measured the root-mean-square and relative errors of the x position of the pendulum bob. The errors in other components of the solution vector were of the same magnitude.

Onestep-like time integration method

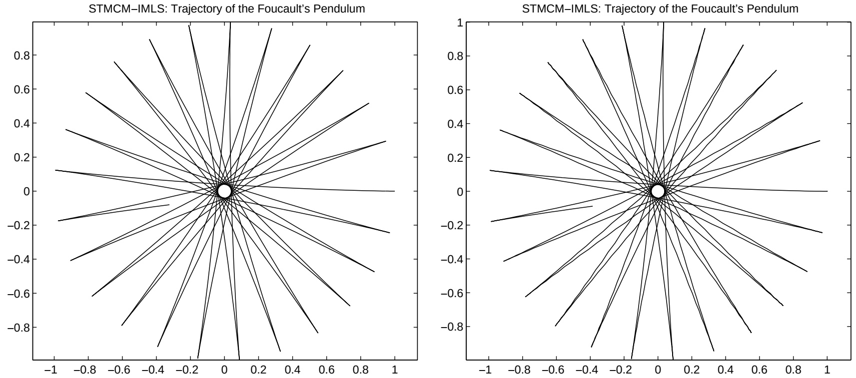
Solution is computed for each of the time slabs of thickness δ_n discretized with 3 Halton points sequentially. Trajectories of the pendulum bob projected onto the floor computed by a onestep-like time integration method via STMCM are given in Figure 6.8 on the next page (*left column*). For the rms-error we obtained the second and for the relative error the third order of convergence.



(a) $\delta_n = 2$ (b) $N = 129$

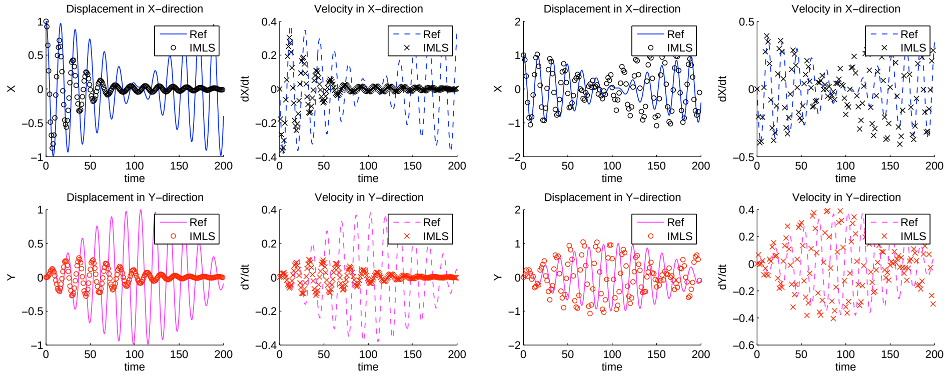


(c) $\delta_n = 1$ (d) $N = 257$

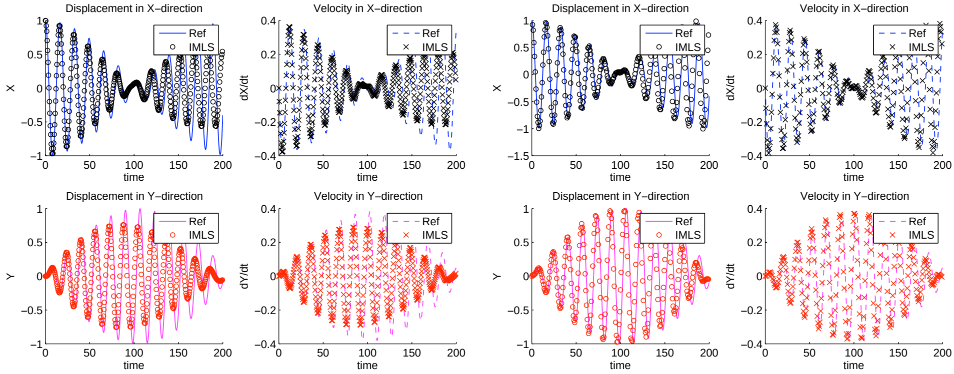


(e) $\delta_n = 0.25$ (f) $N = 1025$

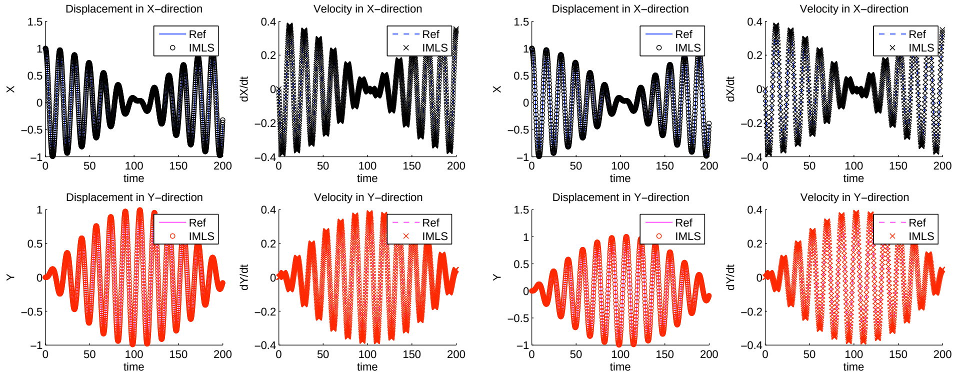
Figure 6.8 | Trajectories of the pendulum bob projected onto the floor computed by a onestep-like (*left column*) and the multistep-like (*right column*) time integration method. The rotation of the Earth is artificially increased to clearly demonstrate the Foucault effect.



(a) $\delta_n = 2$ (b) $N = 129$



(c) $\delta_n = 1$ (d) $N = 257$



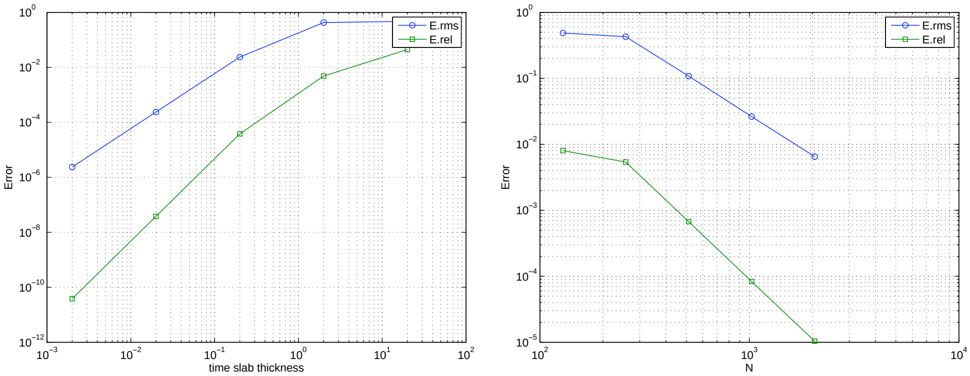
(e) $\delta_n = 0.25$ (f) $N = 1025$

Figure 6.9 | Comparison of the solution convergence of the onestep-like (*left column*) and the multistep-like (*right column*) time integration scheme via STMCM for the Foucault pendulum.

Multistep-like time integration method

The whole time interval is discretized with N Halton points and the solution is computed once. Figure 6.8 on page 84 (*right column*) shows the trajectories of the pendulum bob projected onto the floor computed by a multistep-like time integration method via STMCM. As in the onestep-like method we obtained for the rms-error the second and for the relative error the third order of convergence (see also Figure 6.10).

Comparing the results obtained by both the onestep-like time integration method and the multistep-like one, we see that for this system of linear ordinary equations the similar convergence is obtained. Interestingly, the absolute values of the RMS- and averaged relative errors are lower for the one-step-like solution. However, if the complexity of the system of consideration rises, it is advisable to subdivide the whole time interval into time slabs and obtain the solution subsequently finding the solution for each of the time slabs. Combining the two described approaches gives the STMCM flexibility to choose such parameters as the thickness of the time slab and the number of discretization points. The rule of thumb is: 'the larger thickness of the time slab is chosen, the more points must be used to discretize the time slab.'



(a) *onestep-like* (b) *multistep-like*

Figure 6.10 | Comparison of the convergence of the errors computed for the onestep-like and the multistep-like time integration scheme via STMCM for the Foucault pendulum.

6.3.3 Stability and filter properties

When solving differential equations numerically, the stability of approximation must be assured as consistent and stable approximations lead to convergence. Since for linear ODEs any higher order equation can be written as a system of first order equations, it is sufficient to study equations of the first order.

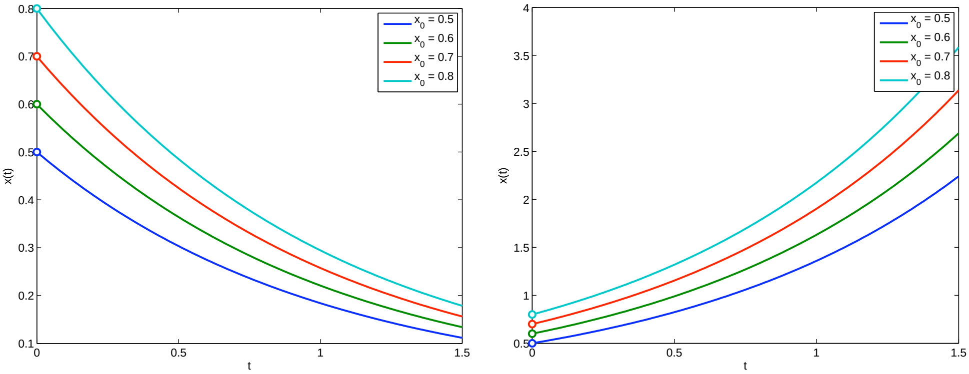
We consider the 1d test equation frequently used to study the stability properties of ODEs

$$\dot{x} = -\mathsf{C}x, \quad \mathsf{C} \text{ const.} \tag{6.13}$$

Analytical solutions to this equation are curves of the form

$$x(t) = x_0 \, \mathrm{e}^{-\mathsf{C}t}, \tag{6.14}$$

where $x_0 = x(t = 0)$ is the initial condition.



(a) $\mathsf{C} > 0$ (b) $\mathsf{C} < 0$

Figure 6.11 | Curve families for a stable (a) and an unstable (b) ODE (6.13) exemplarily for $\mathsf{C} = \pm 1$.

One distinguishes between stable and unstable ODEs depending on their sensitivity to perturbations in initial data. For the test equation (6.13) the sign of C is decisive, see Figure 6.11. A small perturbation of initial data in a stable equation will be damped out, since the curves converge for increasing t; in an unstable equation the initially small perturbations will be amplified due to diverging solution curves.

- ▶ More generally, for an ODE of the form $\dot{x} = f(x,t)$ the sign of eigenvalues of the Jacobian $\mathbf{J} = \left[\frac{\partial f_i}{\partial x_j}\right]$ is important. Since the entries of the Jacobian are functions of t, the stability of the equation may vary with time. If any of the eigenvalues of $\mathbf{J}$ has a positive real part, then the equation is unstable.

In order to study stability of a numerical method for ODEs the equation (6.13) is discretized and rewritten in a form valid for one-step methods

$$x_{n+1} = \mathbf{A} x_n, \tag{6.15}$$

where x_{n+1} and x_n are the solution values at discrete time levels t_{n+1} and t_n, respectively. The matrix $\mathbf{A}$ is called *amplification matrix*.

Spectral radius ρ is defined as the maximum of absolute values of eigenvalues λ_i of the amplification matrix $\mathbf{A}$

$$\rho(\mathbf{A}) = \max_i\{|\lambda_i|\}, \tag{6.16}$$

where $|\lambda_i| = (\lambda_i\overline{\lambda_i})^{1/2}$ with $\overline{\lambda_i}$ being the complex conjugate of λ_i.

► The notation for the spectral radius ρ must not be confused with the notation used previously for the dilatation parameter ρ, nor with another parameter ϱ used in the following section.

Depending on the value of the spectral radius a numerical algorithm has the following properties

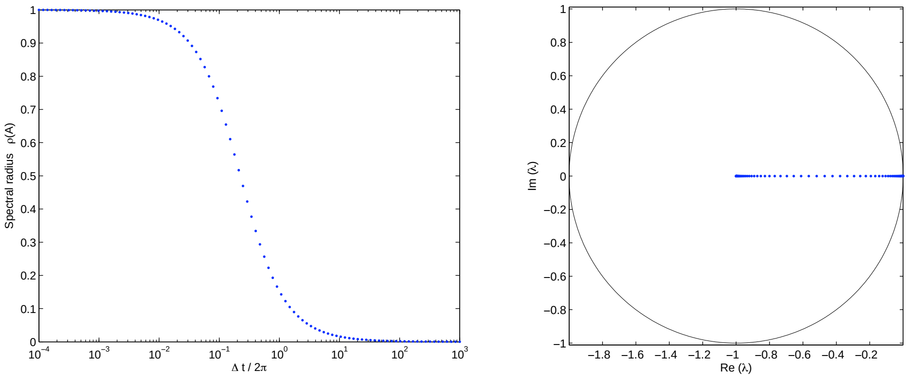
$$\rho(\mathbf{A}) : \begin{cases} \leq 1, & \text{numerical algorithm is stable;} \\ > 1, & \text{numerical algorithm is unstable;} \\ = 1, & \text{no numerical damping;} \\ = 0, & \text{oscillations of the appropriate frequency region are filtered.} \end{cases} \tag{6.17}$$

Spectral radii give a representation of dissipative properties of a numerical algorithm over the entire frequency domain. The magnitude of the spectral radius for the region of high frequencies is of particular interest. For instance for systems of structural dynamics high frequency responses usually arise from the spatial discretization and do not represent the physics. Hence, the high frequency responses should be filtered from the numerical solution by the method. On the other hand, in the region of low frequencies numerical dissipation should be small, otherwise the numerical solution is overdamped, see e.g. [94].

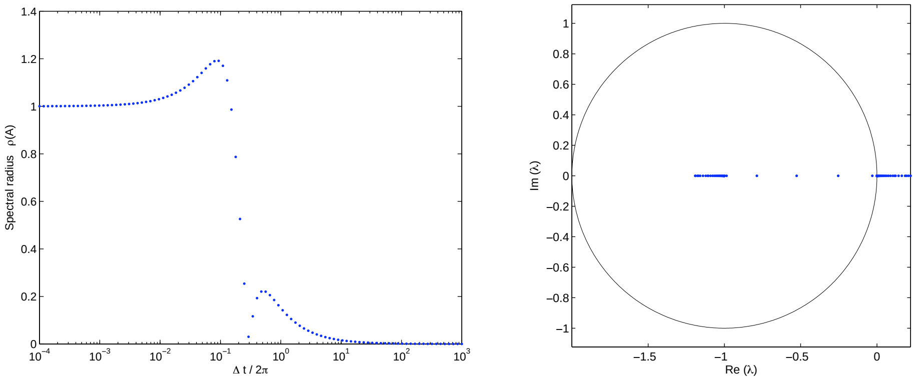
Discretizing (6.13) by STMCM we can compute the eigenvalues of the amplification matrix $\mathbf{A}$, which consists of terms containing derivatives of the IMLS kernel functions w.r.t. time, and the spectral radii ρ for different stepsizes Δt, see Figure 6.12 on the next page. We consider here only time slabs discretized by $N_n = 3$ points and linear basis. A number of algorithms to compute eigenvalues of a matrix are available, e.g. QR-decomposition, Arnoldi or Lanczos iteration (the latter specifically designed for large sparse symmetric matrices), see e.g. [8].

For the stable ODE ($\mathsf{C} > 0$) the method is A-stable, since for all Δt the spectral radius $\rho \leq 1$ and all eigenvalues are real and negative. Lower frequencies are transferred almost undamped, i.e. $\rho \approx 1$. If the thickness of a time slab is increased, the solution is damped numerically, we have already seen this behaviour in the previous section. On the other hand, the higher-frequent oscillations are filtered since $\rho \to 0$ if Δt is increased.

For the unstable ODE ($\mathsf{C} < 0$) the numerical errors are amplified due to the diverging character of the solution curves and the spectral radius $\rho > 1$. If the time step is decreased, the spectral radius is approaching the stability criterion $\rho = 1$ from above. As in the case of a stable ODE, the oscillations of higher frequencies, i.e. for $\Delta t/2\pi > 10$, are also filtered.



(a) $C > 0$



(b) $C < 0$

Figure 6.12 | Spectral radii $\rho(\mathbf{A})$ and eigenvalues λ_i for a STMCM discretization of a stable (a) and an unstable (b) ODE (6.13).

6.4 Nonlinear Ordinary Differential Equations

Most physical phenomena and engineering systems are nonlinear and usually they cannot be solved analytically due to complexity and interaction between the system components. The main difference to linear systems is that the solution of nonlinear ones cannot be represented as a linear combination of solutions of subproblems, i.e. in real-life applications the superposition principle will, most probably, not work. This is also the case in every-day life, e.g. if you watch your favourite movie in two languages simultaneously, by no means will you get double pleasure!

6.4.1 Lorenz attractors

Let us consider a fairly simple-looking coupled system of nonlinear ODEs – the Lorenz equations

$$\begin{aligned} \dot{x} &= \sigma y - x\,, \\ \dot{y} &= \varrho x - y - xz\,, \\ \dot{z} &= xy - \beta z\,, \end{aligned} \tag{6.18}$$

where $\sigma, \varrho, \beta > 0$ are constant parameters, with only two quadratic nonlinear terms in the second and the third equations, namely xz and xy.

The American mathematician and meteorologist Edward N. Lorenz (1917–2008) derived this system from an extremely simplified model of the convection processes in the atmosphere in [136] and his astonishingly comprehensible paper was ahead of its time back in 1963, but became one of the most cited papers in the 1980s – have a look at it!

The parameters in (6.18) have the following physical meanings: σ corresponds to the Prandtl number of the fluid, ϱ is a dimensionless Rayleigh number (the ratio of the temperature difference to damping caused by thermal conductivity and viscosity of the fluid), and β is a geometric factor (the aspect ratio of the convection rolls). This system of equations arises also in models of lasers, dynamos; they describe *exactly* a motion of certain chaotic waterwheels [189], it is possible to build one yourself!

Lorenz discovered that the deterministic system (6.18), though simple-looking, may exhibit a surprisingly chaotic behaviour: for a wide range of parameters σ, ϱ, and β the solutions $x(t), y(t)$, and $z(t)$ oscillate in an irregular fashion. The trajectories never repeat themselves, but always remain on a bounded set called *a strange attractor*. In contrast to stable fixed points, the strange attractor isn't a curve or a surface – it is a fractal with a fractional dimension between two and three, and the motion on it is chaotic. For detailed studies on chaos and nonlinear dynamics systems see e.g. [189, 190].

In the following we shall solve the Lorenz system (6.18) by STMCM (in this case the STMCM reduces to MCM in time) for various parameters, extract the aperiodic motion of the system, and plot the trajectories in three dimensions, obtaining the so called 'butterfly patterns' that are typical for the Lorenz equations. The term 'butterfly effect' appeared due to the shape of the strange attractor trajectory and was used by Lorenz in his paper presented in 1972 entitled: "Predictability: does the flap of a butterfly's wings in Brazil set off a tornado in Texas?" The original Lorenz's parameters $\sigma = 10$ (which is too high for dry air and would correspond to a cold fluid), $\beta = 8/3$, and several values for $\varrho = \{14, 28, 350\}$ are chosen.

Figure 6.13 on the facing page shows the trajectories of the Lorenz equations for these parameters computed via STMCM (linear basis, dilatation parameter $\rho = 3 \cdot h_{\Omega,\mathcal{S}}$, the parameter of the weight function $\alpha = 6$, time slab thickness $\delta_n = 0.01$, and number of Halton points to discretize each time slab N_n=17) with initial conditions $x_0 = z_0 = 0$, $y_0 = 1$. The nonlinearities were resolved via Picard fixed point iteration scheme as described in Section 3.3 on page 38.


(a) $\varrho = 14$ (b) $\varrho = 28$ (c) $\varrho = 350$

Figure 6.13 | Typical trajectories for Lorenz equations with $\sigma = 10$, $\beta = 8/3$ solved by STMCM. Note the fixed point (a), the butterfly pattern (b), and the approach to the final attracting limit cycle (c).

For $\varrho < 13.926$ there are no limit cycles [189]. A fixed point solution is obtained for $\varrho = 14$, as shown in Figure 6.13(a). A critical value for the Lorenz equations is $\varrho = 24.06$ [189]. Beyond this value, the set becomes a strange attractor and the butterfly pattern appears, see Figure 6.13(b). An initially chaotic change of stable fixed points ultimately dampes down to an equilibrium on the right, but a slight change of initial conditions results in an unpredictable behaviour of this deterministic system (it is not possible to predict at which of the two saddle points the solution eventually will arrive, at least for a certain range of initial conditions). This behaviour, also called *transient chaos*, shows that a deterministic system, even if its final state is very simple, may in fact be unpredictable. There is a certain resemblance of this behaviour to natural phenomena. For instance, when rolling dice the prediction of the final position is difficult, since the result depends sensitively on the initial orientation, velocity, table material, etc. But eventually the die will stop in one of the six stable equilibrium positions, the same considerations are also valid for a roulette game in a casino or the "Russian roulette". As proved in [185], for all $\varrho > 313$ the solution of the Lorenz system has a final globally attracting limit cycle, which is demonstrated also by our solution for $\varrho = 350$, see Figure 6.13(c).

A general closed form solution for the Lorenz system is not available. Therefore, we compare the results with the solution obtained by an explicit multistep Adams-Bashforth-Moulton predictor-corrector (ABM-PECE) scheme of 13th order with absolute and relative tolerances 10^{-9}. Time histories of solutions by STMCM $x(t), y(t), z(t)$ are compared to those by the reference integration method in Figure 6.14 on page 93 for different parameters ϱ, whereas in the left column each time slab was discretized by N_n=17 and in the right column by N_n=513 Halton points. In both cases all other STMCM parameters were kept constant at the values mentioned above. For the first case of low density of points used for discretization of the time slabs our solution exhibits a phase error compared to the reference solution. Yet qualitatively, it does represent the expected behaviour: a point with coordinates (x, y, z) remains within the bounded strange attractor, although the exact position is different from the one computed by the reference method. It can be clearly seen that the phase error reduces and the STMCM is in excellent agreement with the reference solution, if the density of time slab discretization points is increased (see Figure 6.14b).

Computational errors between the STMCM solution and the reference as well as the experimental orders of convergence, computed according the error indicators from Section 5.1 for the whole time interval, are presented exemplarily for the solution $y(t)$ in Table 6.2. In all three cases we observe good convergence rates: for parameter ϱ leading to a fixed point solution and a solution with a limit cycle the expected second order convergence for averaged relative and maximum errors and the first order convergence for the root-mean-square error; for the case of a strange attractor the convergence is of second order for the RMS error and approaches the third order for averaged relative and absolute errors.

Table 6.2 | Convergence study of the solution of Lorenz system (6.18) by STMCM.

N_n	$\mathrm{E}_{\mathrm{rms}}$	$\mathrm{E}_{\mathrm{rel}}$	$\mathrm{E}_{\mathrm{abs}}$	$\mathrm{EOC}_{\mathrm{rms}}$	$\mathrm{EOC}_{\mathrm{rel}}$	$\mathrm{EOC}_{\mathrm{abs}}$
		(a)	$\varrho = 14$			
17	1.8223e+00	1.3798e-05	1.8971e-04	–	–	–
257	1.3842e-01	7.1330e-08	9.8069e-07	0.9491	1.9386	1.9386
513	6.9064e-02	1.7843e-08	2.4532e-07	1.0058	2.0048	2.0048
		(b)	$\varrho = 28$			
17	9.4875e+00	2.6226e-05	7.2634e-04	–	–	–
257	6.5993e+00	1.5000e-06	4.1543e-05	0.1337	1.0536	1.0536
513	1.6301e+00	2.3504e-07	5.8069e-06	2.0230	2.6815	2.8468
		(c)	$\varrho = 350$			
17	8.3995e+01	4.9750e-05	1.7370e-02	–	–	–
257	3.4385e+01	1.4035e-06	4.9007e-04	0.3289	1.3138	1.3138
513	1.8180e+01	4.0610e-07	1.4180e-04	0.9221	1.7942	1.7942

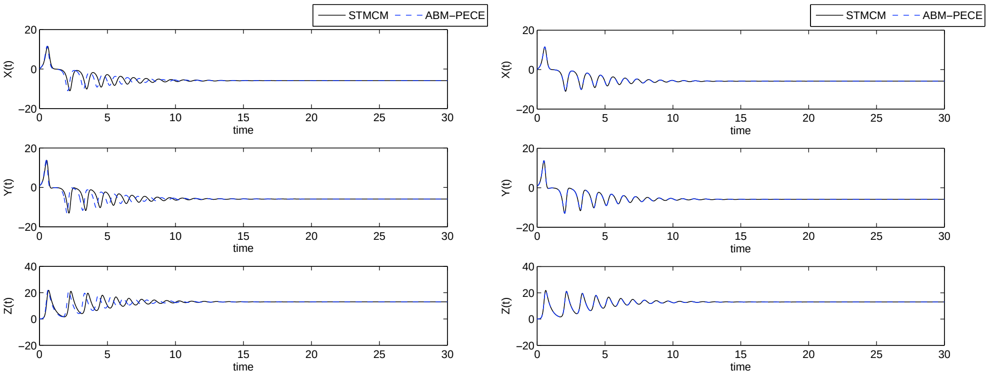
Let us now proceed further and apply the computational method proposed in Chapter 3 on page 35 to the numerical solution of various types of partial differential equations.

6.5 Stationary Problems :: Elliptic Partial Differential Equations

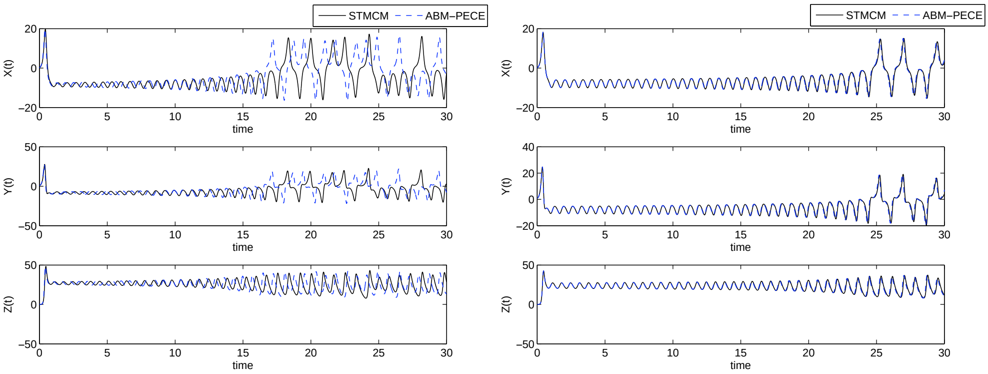
Mathematical modelling of stationary or steady-state processes, e.g. constant temperature distribution, deals with partial differential equations of elliptic type leading to boundary value problems (BVPs).The theory of general BVPs can be found e.g. in [135] and the numerical treatment of elliptic problems by classical methods is available, see e.g. [79]. We apply in this section the STMCM presented in Chapter 3 on page 35 which, for elliptic problems, reduces to a MCM in space only since there is no time-dependency.

Without loss of generality we consider an elliptic problem – the Poisson's equation

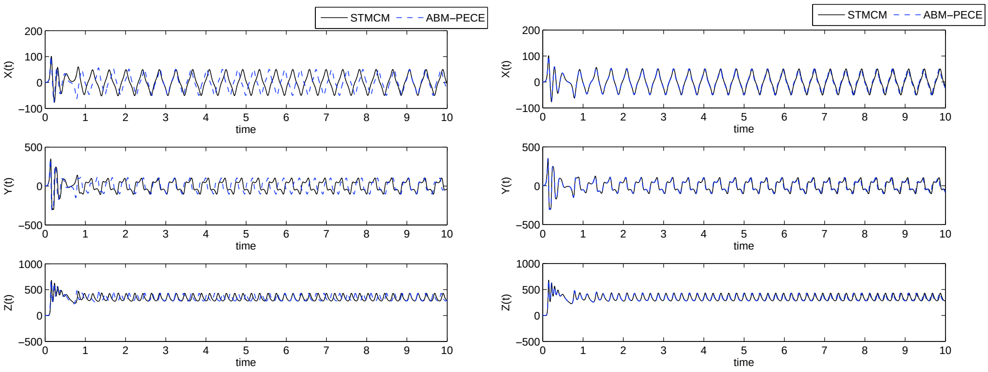
$$\Delta \mathrm{u} = \mathrm{f}_{\Omega} \quad \text{in} \quad \Omega \subset \mathbb{R}^d \tag{6.19}$$



(a) $\varrho = 14$



(b) $\varrho = 28$



(c) $\varrho = 350$

Figure 6.14 | Comparison of time histories for Lorenz equations (6.18) solved by STMCM and ABM-PECE for standard Lorenz parameters. A single time slab of thickness $\delta_n = 0.01$ in STMCM is discretized by $N_n = 17$ (*left column, phase error*) and $N_n = 513$ (*right column, good agreement*) Halton points.

with the Dirichlet boundary conditions

$$\mathbf{u} = \mathbf{f}_{\Gamma_\mathrm{D}} \quad \text{on} \quad \Gamma_D \tag{6.20}$$

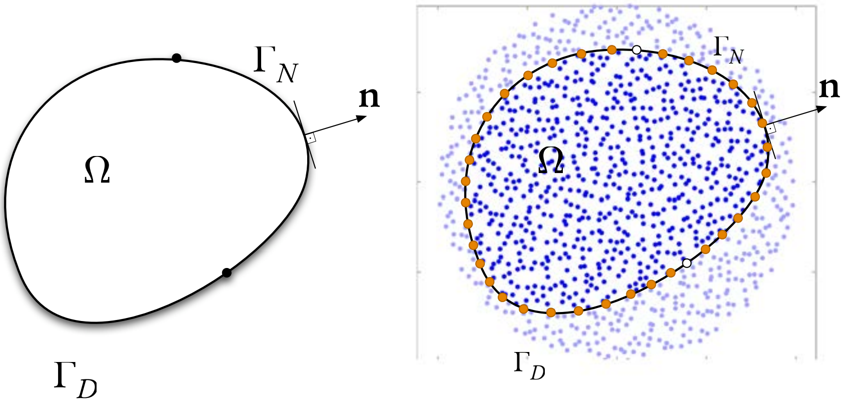
and the von Neumann boundary conditions

$$\frac{\partial \mathbf{u}}{\partial \mathbf{n}} = \mathbf{f}_{\Gamma_\mathrm{N}} \quad \text{on} \quad \Gamma_N. \tag{6.21}$$

The Laplacian operator is defined as $\Delta := \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$, where $\mathbf{x} = (x_1, x_2, \ldots, x_d)^\mathsf{T} \in \mathbb{R}^d$ are the arguments of $\mathbf{u}, \mathbf{f}_\Omega, \mathbf{f}_{\Gamma_\mathrm{D}}$ and $\mathbf{f}_{\Gamma_\mathrm{N}}$.

Ω is an open domain of the d-dimensional space. Its boundary $\Gamma := \partial\Omega = \Gamma_N \cup \Gamma_D$ is a union of the Dirichlet and von Neumann boundaries Γ_D and Γ_N, respectively (see Figure 6.15), and $\frac{\partial \mathbf{u}}{\partial \mathbf{n}}$ is the directional derivative w.r.t. the outer normal $\mathbf{n}$ at the point $\mathbf{x} \in \Gamma_N$.

We use Halton points (see Section 3.4.1 on page 39) augmented with a discretization of a boundary Γ, overlaying the domain of interest Ω with a bigger circle, and then abandoning the points lying beyond the boundary Γ according to the implicit strategy described in Chapter 4.



(a) *general* (b) *discretization*

Figure 6.15 | Computational domain for a boundary value problem.

Approximated solution $u^{\mathrm{appx}}(\mathbf{x})$ of the BVP (6.19) with the boundary conditions (6.20)–(6.21) is sought in the following form:

$$u^{\mathrm{appx}}(\mathbf{x}) := \sum_{i=1}^{N} u_i \cdot \varphi_i(\mathbf{x}) = < \mathbf{u}, \boldsymbol{\varphi}(\mathbf{x}) >, \tag{6.22}$$

where $\mathbf{u} = (u_1, u_2, \ldots, u_N)^\mathsf{T}$ is a vector of the solution values at N data points located at $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and $\boldsymbol{\varphi}(\mathbf{x}) = (\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), \ldots, \varphi_N(\mathbf{x}))^\mathsf{T}$ is a vector of ansatz IMLS functions evaluated at the point $\mathbf{x}$, $< \cdot, \cdot >$ represents a scalar product of two vectors.

6.5.1 Theoretical issues

In this section we present some theoretical issues which are of importance for the following collocation solution of boundary value problems, especially the ones with von Neumann boundary conditions.

Definition 6.3 *A $N \times N$ complex matrix $\mathbf{A} = [a_{i,j}]$, $(i, j = 1 \ldots N)$ is called* strictly diagonally dominant *if*

$$\left|a_{i,i}\right| > \sum_{\substack{j=1 \\ j \neq i}}^{N} \left|a_{i,j}\right|, \quad \forall\, i : 1 \leqslant i \leqslant N.$$

Lemma 6.4 *Let $\mathbf{A} = [a_{i,j}]$, $(i, j = 1 \ldots N)$ be a $N \times N$ diagonally dominant complex matrix. Then, a $n \times n$ submatrix $\mathbf{B} = [a_{k,l}]$, $(k, l = m \ldots (n - m + 1) : 1 \leqslant m \leqslant n \leqslant N)$ is also diagonally dominant.*

Proof Obvious (see Figure 6.16). ☐

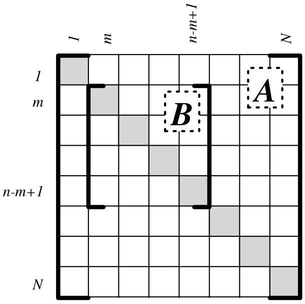


Figure 6.16 | Diagonally dominant matrix **A** and its submatrix **B**.

Lemma 6.5 *Let $\mathbf{A} = [a_{i,j}]$, $(i, j = 1 \ldots N)$ be an $N \times N$ strictly diagonally dominant complex matrix. Then, the matrix $\mathbf{A}$ is nonsingular.*

Proof For a detailed proof see [196], [210]. ☐

Theorem 6.6 *Let* $\mathscr{S} = \mathscr{Q} \cup \mathscr{G} := \{1, \ldots, N\}$ *be an index set of all points in a computation domain, so that* $\mathscr{G} := \{k, \ldots, l\}$ *is an index set of points on the von Neumann boundary and* $\mathscr{Q} := \{1 \ldots, k-1, l+1, \ldots, N\}$ *an index set of all other points.*

Let $\mathscr{D} := \frac{\partial}{\partial \mathbf{n}}$ *be a derivative operator in the direction of an outer normal vector* **n** *and* $\varphi(x)$ *- the IMLS kernel functions.*

If for all $i \in \mathscr{G}$ *the condition*

$$\left| \sum_{j \in \mathscr{Q}} \mathscr{D}\varphi_j(x_i) \right| > 2 \sum_{j \in \mathscr{G} \setminus \{i\}} \left| \mathscr{D}\varphi_j(x_i) \right| \tag{6.23}$$

is fulfilled, then it holds:

$$\forall i \in \mathscr{G}: \qquad |\mathscr{D}\varphi_i(x_i)| > \sum_{j \in \mathscr{G} \setminus \{i\}} \left| \mathscr{D}\varphi_j(x_i) \right|. \tag{6.24}$$

Proof The partition of nullity for the derivatives of the kernel functions (see Section 2.6.3 on page 23) writes as

$$\forall i \in \mathscr{S}: \qquad \sum_{j \in \mathscr{S}} \mathscr{D}\varphi_j(x_i) = 0$$

Hence $\forall i \in \mathscr{G}$ it also holds

$$\sum_{j \in \mathscr{S}} \mathscr{D}\varphi_j(x_i) = 0,$$

which, since $\mathscr{S} = \mathscr{Q} \cup \mathscr{G}$, can be rewritten as

$$\mathscr{D}\varphi_i(x_i) + \sum_{j \in \mathscr{Q}} \mathscr{D}\varphi_j(x_i) + \sum_{j \in \mathscr{G} \setminus \{i\}} \mathscr{D}\varphi_j(x_i) = 0 \qquad \Longrightarrow$$
$$-\mathscr{D}\varphi_i(x_i) = \sum_{j \in \mathscr{Q}} \mathscr{D}\varphi_j(x_i) + \sum_{j \in \mathscr{G} \setminus \{i\}} \mathscr{D}\varphi_j(x_i).$$

Applying the modulus operator on both sides of the latter equation we obtain

$$|-\mathscr{D}\varphi_i(x_i)| = |\mathscr{D}\varphi_i(x_i)| = \left| \sum_{j \in \mathscr{Q}} \mathscr{D}\varphi_j(x_i) + \sum_{j \in \mathscr{G} \setminus \{i\}} \mathscr{D}\varphi_j(x_i) \right|,$$

and with the help of the triangle inequality as well as the condition (6.23) we get

$$\begin{aligned}|\mathscr{D}\varphi_i(x_i)| \;=\; & \left|\sum_{j\in\mathscr{Q}}\mathscr{D}\varphi_j(x_i)+\sum_{j\in\mathscr{G}\setminus\{i\}}\mathscr{D}\varphi_j(x_i)\right| \geqslant \\ & \left|\sum_{j\in\mathscr{Q}}\mathscr{D}\varphi_j(x_i)\right| - \left|\sum_{j\in\mathscr{G}\setminus\{i\}}\mathscr{D}\varphi_j(x_i)\right| \geqslant \\ & \left|\sum_{j\in\mathscr{Q}}\mathscr{D}\varphi_j(x_i)\right| - \sum_{j\in\mathscr{G}\setminus\{i\}}\left|\mathscr{D}\varphi_j(x_i)\right| > \\ & 2\sum_{j\in\mathscr{G}\setminus\{i\}}\left|\mathscr{D}\varphi_j(x_i)\right| - \sum_{j\in\mathscr{G}\setminus\{i\}}\left|\mathscr{D}\varphi_j(x_i)\right| = \\ & \sum_{j\in\mathscr{G}\setminus\{i\}}\left|\mathscr{D}\varphi_j(x_i)\right|. \qquad \square\end{aligned}$$

6.5.2 Discretization and boundary conditions

Now we discretize the BVP (6.19) using the ansatz from Section 6.5. We subdivide the vector of the solution into three subvectors:

$$\mathbf{u} = (\underbrace{u_1,\ldots,u_{k-1}}_{\mathbf{u}_\Omega},\underbrace{u_k,\ldots,u_l}_{\mathbf{u}_{\Gamma_\mathrm{N}}},\underbrace{u_{l+1},\ldots,u_N}_{\mathbf{u}_{\Gamma_\mathrm{D}}})^\mathsf{T}, \tag{6.25}$$

where the indices $k \leqslant l \leqslant N$ have the following meaning:

- $1..(k-1)$ refer to the points within the domain Ω,
- $k..l$ refer to the points on the von Neumann boundary Γ_N,
- $(l+1)..N$ refer to the points on the Dirichlet boundary Γ_D.

A discretization of the left-hand side of (6.19) for all $\mathbf{x}\in\Omega$ writes as

$$\Delta u(\mathbf{x}) \approx \Delta u^{\mathrm{appx}}(\mathbf{x}) = \sum_{i=1}^{N}\Delta\varphi_i(\mathbf{x})\cdot u_i. \tag{6.26}$$

Rewriting the solution vector $\mathbf{u}$ from (6.25) as a block vector

$$\mathbf{u} = (\mathbf{u}_\Omega \mid \mathbf{u}_{\Gamma_\mathrm{N}} \mid \mathbf{u}_{\Gamma_\mathrm{D}})^\mathsf{T} \in \mathbb{R}^N \tag{6.27}$$

and the discretized Laplacian as a block matrix

$$\mathscr{L} = \left(\mathscr{L}_\Omega \mid \mathscr{L}_{\Gamma_\mathrm{N}} \mid \mathscr{L}_{\Gamma_\mathrm{D}}\right) \in \mathbb{R}^{(k-1)\times N}, \tag{6.28}$$

where

$$\mathscr{L}_\Omega = \begin{pmatrix} \Delta\varphi_1(\mathbf{x}_1) & \dots & \Delta\varphi_{k-1}(\mathbf{x}_1) \\ \Delta\varphi_1(\mathbf{x}_2) & \dots & \Delta\varphi_{k-1}(\mathbf{x}_2) \\ \vdots & \ddots & \vdots \\ \Delta\varphi_1(\mathbf{x}_{k-1}) & \dots & \Delta\varphi_{k-1}(\mathbf{x}_{k-1}) \end{pmatrix} \in \mathbb{R}^{(k-1)\times(k-1)},$$

$$\mathscr{L}_{\Gamma_\mathrm{N}} = \begin{pmatrix} \Delta\varphi_k(\mathbf{x}_1) & \dots & \Delta\varphi_l(\mathbf{x}_1) \\ \Delta\varphi_k(\mathbf{x}_2) & \dots & \Delta\varphi_l(\mathbf{x}_2) \\ \vdots & \ddots & \vdots \\ \Delta\varphi_k(\mathbf{x}_{k-1}) & \dots & \Delta\varphi_l(\mathbf{x}_{k-1}) \end{pmatrix} \in \mathbb{R}^{(k-1)\times(l-k+1)},$$

and

$$\mathscr{L}_{\Gamma_\mathrm{D}} = \begin{pmatrix} \Delta\varphi_{l+1}(\mathbf{x}_1) & \dots & \Delta\varphi_N(\mathbf{x}_1) \\ \Delta\varphi_{l+1}(\mathbf{x}_2) & \dots & \Delta\varphi_N(\mathbf{x}_2) \\ \vdots & \ddots & \vdots \\ \Delta\varphi_{l+1}(\mathbf{x}_{k-1}) & \dots & \Delta\varphi_N(\mathbf{x}_{k-1}) \end{pmatrix} \in \mathbb{R}^{(k-1)\times(N-l)},$$

we get the discretization of the equation (6.19) in a matrix form as

$$\mathscr{L} \cdot \mathbf{u} = \mathbf{f}_\Omega, \tag{6.29}$$

with $\mathbf{f}_\Omega = (f_1, \dots, f_{k-1})^\mathsf{T} \in \mathbb{R}^{k-1}$.

Using the structures of $\mathscr{L}$ and $\mathbf{u}$ in (6.28) and (6.27), respectively, we can rewrite equation (6.29) as

$$\mathscr{L}_\Omega \cdot \mathbf{u}_\Omega = \mathbf{f}_\Omega - \mathscr{L}_{\Gamma_\mathrm{D}} \cdot \mathbf{u}_{\Gamma_\mathrm{D}} - \mathscr{L}_{\Gamma_\mathrm{N}} \cdot \mathbf{u}_{\Gamma_\mathrm{N}}. \tag{6.30}$$

The Dirichlet boundary conditions are given from (6.20) as $\mathbf{u}_{\Gamma_\mathrm{D}} = \mathbf{f}_{\Gamma_\mathrm{D}}$.

The von Neumann boundary conditions from (6.21) write as $\frac{\partial}{\partial \mathbf{n}} \mathbf{u}_{\Gamma_\mathrm{N}} = \mathbf{f}_{\Gamma_\mathrm{N}}$.

Hence, taking use of the ansatz for derivatives of the kernel functions from Section 2.6.3 on page 23, for all $\mathbf{x} \in \Gamma_N$ we obtain:

$$\frac{\partial}{\partial \mathbf{n}} u(\mathbf{x}) \approx \frac{\partial}{\partial \mathbf{n}} u^{\mathrm{appx}}(\mathbf{x}) = \sum_{i=1}^{N} \frac{\partial \varphi_i(\mathbf{x})}{\partial \mathbf{n}} \cdot u_i = f_{\Gamma_N}$$

or in matrix form

$$\mathscr{D}\mathbf{\Phi} \cdot \mathbf{u} = \mathbf{f}_{\Gamma_\mathrm{N}}, \tag{6.31}$$

where

$$\mathscr{D}\mathbf{\Phi} = \frac{\partial}{\partial \mathbf{n}} \begin{pmatrix} \varphi_1(\mathbf{x}_k) & \varphi_2(\mathbf{x}_k) & \dots & \varphi_N(\mathbf{x}_k) \\ \varphi_1(\mathbf{x}_{k+1}) & \varphi_2(\mathbf{x}_{k+1}) & \dots & \varphi_N(\mathbf{x}_{k+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(\mathbf{x}_{l-1}) & \varphi_2(\mathbf{x}_{l-1}) & \dots & \varphi_N(\mathbf{x}_{l-1}) \\ \varphi_1(\mathbf{x}_l) & \varphi_2(\mathbf{x}_l) & \dots & \varphi_N(\mathbf{x}_l) \end{pmatrix} \in \mathbb{R}^{(l-k+1)\times N}. \tag{6.32}$$

And again we can write the matrix (6.32) as a block matrix

$$\mathscr{D}\boldsymbol{\Phi} = \left(\mathscr{D}\boldsymbol{\Phi}_\Omega \mid \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}} \mid \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{D}}\right), \tag{6.33}$$

where

$$\mathscr{D}\boldsymbol{\Phi}_\Omega = \frac{\partial}{\partial \mathbf{n}} \begin{pmatrix} \varphi_1(\mathbf{x}_k) & \dots & \varphi_{k-1}(\mathbf{x}_k) \\ \varphi_1(\mathbf{x}_{k+1}) & \dots & \varphi_{k-1}(\mathbf{x}_{k+1}) \\ \vdots & \ddots & \vdots \\ \varphi_1(\mathbf{x}_l) & \dots & \varphi_{k-1}(\mathbf{x}_l) \end{pmatrix} \in \mathbb{R}^{(l-k+1)\times(k-1)},$$

$$\mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}} = \frac{\partial}{\partial \mathbf{n}} \begin{pmatrix} \varphi_k(\mathbf{x}_k) & \dots & \varphi_l(\mathbf{x}_k) \\ \varphi_k(\mathbf{x}_{k+1}) & \dots & \varphi_l(\mathbf{x}_{k+1}) \\ \vdots & \ddots & \vdots \\ \varphi_k(\mathbf{x}_l) & \dots & \varphi_l(\mathbf{x}_l) \end{pmatrix} \in \mathbb{R}^{(l-k+1)\times(l-k+1)}, \tag{6.34}$$

$$\mathscr{D}\boldsymbol{\Phi}_\Omega = \frac{\partial}{\partial \mathbf{n}} \begin{pmatrix} \varphi_{l+1}(\mathbf{x}_k) & \dots & \varphi_N(\mathbf{x}_k) \\ \varphi_{l+1}(\mathbf{x}_{k+1}) & \dots & \varphi_N(\mathbf{x}_{k+1}) \\ \vdots & \ddots & \vdots \\ \varphi_{l+1}(\mathbf{x}_l) & \dots & \varphi_N(\mathbf{x}_l) \end{pmatrix} \in \mathbb{R}^{(l-k+1)\times(N-l)}.$$

Now we rewrite the equation (6.31) and get

$$\mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}} \cdot \mathbf{u}_{\Gamma_\mathrm{N}} = \mathbf{f}_{\Gamma_\mathrm{N}} - \mathscr{D}\boldsymbol{\Phi}_\Omega \cdot \mathbf{u}_\Omega - \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{D}} \cdot \mathbf{u}_{\Gamma_\mathrm{D}}. \tag{6.35}$$

It can be shown [148] that the square matrix (6.34) is diagonally dominant, and hence we can apply Theorem 6.6 from the Section 6.5.1 and constitute that this matrix $\mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}$ is nonsingular and there exists an inverse $\mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1}$.

- ▶ Condition (6.23) refers to the well-posedness of the problem in the sense described in Section 6.2.2 on page 75, as well as to the positivity conditions as described in Section 2.8.2 on page 34.

Hence we write

$$\mathbf{u}_{\Gamma_\mathrm{N}} = \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \left[\mathbf{f}_{\Gamma_\mathrm{N}} - \mathscr{D}\boldsymbol{\Phi}_\Omega \cdot \mathbf{u}_\Omega - \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{D}} \cdot \mathbf{u}_{\Gamma_\mathrm{D}}\right]. \tag{6.36}$$

Inserting (6.36) in (6.30) and rearranging we write

$$\begin{aligned} &[\,\mathscr{L}_\Omega - \mathrm{L}_{\Gamma_\mathrm{N}} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \mathscr{D}\boldsymbol{\Phi}_\Omega\,] \cdot \mathbf{u}_\Omega = \\ &\qquad \mathbf{f}_\Omega - \mathscr{L}_{\Gamma_\mathrm{D}} \cdot \mathbf{u}_{\Gamma_\mathrm{D}} - \mathscr{L}_{\Gamma_\mathrm{N}} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \left[\mathbf{f}_{\Gamma_\mathrm{N}} - \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{D}} \mathbf{u}_{\Gamma_\mathrm{D}}\right], \end{aligned}$$

and finally obtain the linear system of equations on $\mathbf{u}_\Omega$:

$$\begin{aligned} &[\,\mathscr{L}_\Omega - \mathscr{L}_{\Gamma_\mathrm{N}} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \mathscr{D}\boldsymbol{\Phi}_\Omega\,] \cdot \mathbf{u}_\Omega = \\ &\mathbf{f}_\Omega - [\,\mathscr{L}_{\Gamma_\mathrm{D}} - \mathscr{L}_{\Gamma_\mathrm{N}} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{D}}\,] \cdot \mathbf{u}_{\Gamma_\mathrm{D}} - \mathscr{L}_{\Gamma_\mathrm{N}} \cdot \mathscr{D}\boldsymbol{\Phi}_{\Gamma_\mathrm{N}}^{-1} \cdot \mathbf{f}_{\Gamma_\mathrm{N}}. \end{aligned} \tag{6.37}$$

So, solving the BVP (6.19) with the boundary conditions (6.20)–(6.21) we proceed by solving the system of equations (6.37) for $\mathbf{u}_\Omega$ and afterwards computing the values on the von Neumann boundary $\mathbf{u}_{\Gamma_N}$ from (6.36).

- ▶ If boundary conditions of different types have to be applied to the same part of the domain boundary, e.g. both Dirichlet and Neumann boundary conditions, then these conditions must be introduced as additional equations. Although this will increase the problem size, collocation methods are known to give more accurate results in this case [59].

6.5.3 Laplace equation on a unit square

Consider the following boundary value problem

$$\Delta u = 0 \tag{6.38}$$

on $0 < x_1, x_2 < 1$ with the boundary values

$$\begin{aligned} u(x_1,0) &= \sin(\pi x_1) \\ u(0,x_2) &= \sin(\pi x_2) \\ u(1,0) &= u(0,1) = 0. \end{aligned} \tag{6.39}$$

The exact solution to this problem is known:

$$u(\mathbf{x}) = \frac{\sinh(\pi(1-x_1))}{\sinh \pi}\sin(\pi x_2) + \frac{\sinh(\pi(1-x_2))}{\sinh \pi}\sin(\pi x_1),$$

which gives us the possibility to compute the error between the approximated and the exact solutions. A quadratic basis and N uniformly distributed points in the domain with a meshsize h were used for this example.

The convergence results presented in Table 6.3 show good agreement with the analytical solution and for both chosen values of the exponent α of the weight function (2.18) the expected third order of convergence is obtained.

6.5.4 Laplace equation on a circle

Let us consider the Laplace equation on a circle with radius $r : 0 < r < R$

$$\Delta u(r,\theta) = 0, \quad -\pi < \theta < \pi \tag{6.40}$$

with prescribed boundary conditions

$$u(r = R) = f(\theta) := \sin(k\theta), \quad k \in \mathbb{N}, \tag{6.41}$$

Table 6.3 | Solution of the boundary value problem (6.38)–(6.39).

N	h	$\mathsf{E}_{\mathsf{rel}}\vert_{\alpha=2}$	$\mathsf{EOC}_{\mathsf{rel}}\vert_{\alpha=2}$	$\mathsf{E}_{\mathsf{rel}}\vert_{\alpha=4}$	$\mathsf{EOC}_{\mathsf{rel}}\vert_{\alpha=4}$
25	0.2500e+00	1.7725e-1	—	7.7221e-02	—
64	1.4286e-01	4.7782e-02	2.3425	2.3404e-02	2.1332
100	1.1111e-01	2.3430e-02	2.8356	1.3525e-02	2.1820
225	7.1429e-02	6.4265e-03	2.9278	5.1794e-03	2.1724
400	5.2632e-02	2.5979e-03	2.9659	9.2167e-04	5.6528
625	4.1667e-02	1.2955e-03	2.9784	4.5974e-04	2.9772
841	3.5714e-02	8.0568e-04	3.0812	2.9784e-04	2.8161

Table 6.4 | Convergence study of the Laplace equation on a circle (6.40)–(6.41) for $k = 5$.

$h_{\Omega,\mathscr{S}}$	$\mathsf{E}_{\mathsf{rel}}$	$\mathsf{EOC}_{\mathsf{rel}}$
4.358987e-01	1.1111e-02	-
2.666864e-01	3.1450e-03	2.5688
1.271675e-01	8.3478e-04	1.7911
7.464144e-02	2.1516e-04	2.5446

which describes a vibration of a circular membrane.

The exact solution to this problem is given by

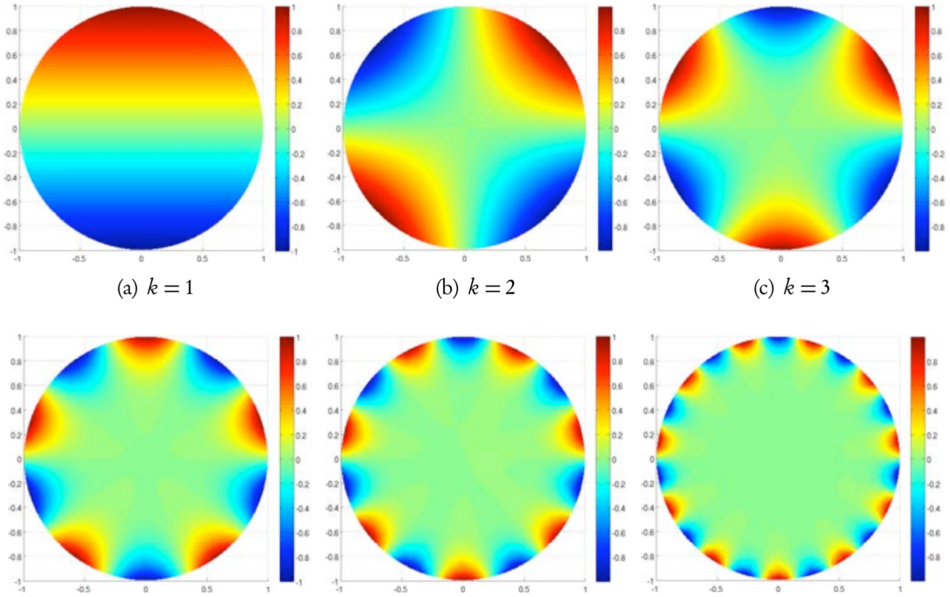
$$u^{\mathsf{ex}}(r,\theta) = r^k \sin(k\theta). \tag{6.42}$$

Figure 6.17 on the next page shows the solutions to (6.40) with varying parameter k in boundary conditions (6.41) computed on a set of $N = 1089$ Halton points augmented with $N_{\mathrm{b}} = 100$ points uniformly distributed on a boundary ($R = 1$). Numerical solutions are in excellent agreement with the analytical ones (6.42). The characteristic behaviour of the solution, i.e. prescribed oscillations on a boundary and flat region towards the center of the computational domain, is resolved very well.

The relative errors between the numerical solution by STMCM and the exact solution in the computational domain for the case $k = 5$ are given in Table 6.4, the method is of third order convergence.

6.5.5 A boundary layer problem

A boundary layer phenomenon, discovered by the German physicist Ludwig Prandtl (1875–1953) in 1904, appears in many physical systems. For instance, the atmosphere exhibits boundary layer structure near the surface of the Earth due to heat radiation and moisture



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

(d) $k = 5$ (e) $k = 7$ (f) $k = 11$

Figure 6.17 | Numerical solution by STMCM of a Laplace equation on a unit circle ($R = 1$) with prescribed boundary conditions (6.40)–(6.41) with varying parameter k.

conditions (also called 'planetary boundary layer' or 'atmospheric boundary layer'), in aerodynamics in order to correctly resolve the flow around the wing of an aeroplane one must consider the boundary layer phenomenon due to the fluid viscosity, etc.

This section discusses the accuracy of the STMCM when applied to a boundary layer problem with a known analytical solution. Breuß et al. consider in [27] this problem and solve the heat equation by a finite volume technique with an anisotropic mesh adapted in the vicinity of the boundary to capture the steep gradients.

The equation under consideration is the heat equation

$$\partial_t \theta(\mathbf{x}, t) = \nabla \cdot [\lambda(\mathbf{x}) \nabla \theta(\mathbf{x}, t)] + q(\mathbf{x}, t), \tag{6.43}$$

where θ is temperature, q is the heat source, λ is the coefficient of thermal conductivity. Of course, a set of initial and boundary conditions must be specified for this initial boundary value problem.

In general, there is no analytical solution to boundary layer problems. However, for a simplified two-dimensional case, i.e. $\mathbf{x} = (x, y)^{\mathsf{T}}$ with $\lambda = 1$ and a specific choice of the time-independent heat source

$$q(\mathbf{x}, t) = q(x) = \gamma^2 \cdot x^{\gamma - 2}, \quad \gamma = 2k, \; k \in \mathbb{N} \tag{6.44}$$

the steady state solution of problem (6.43) is

$$\theta(x) = 2 - x^{\gamma}, \tag{6.45}$$

which is shown in Figure 6.18 for different parameters γ of the source term (6.44). As γ increases, the solution in the immediate vicinity of the boundary exhibits steep gradients, whereas within the domain the gradient is approaching zero as $\gamma \to \infty$. Moreover, for higher values of γ the thickness of the boundary layer, i.e. the area with steep gradients, gets thinner.

The *parabolic* PDE (6.44) reduces in steady state to a boundary value problem of elliptic type, namely to the Laplace equation.

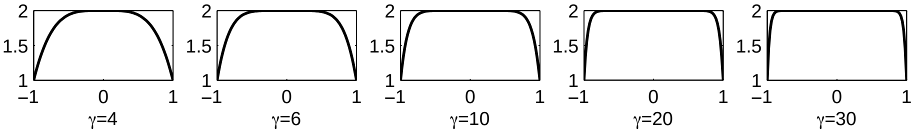


Figure 6.18 | Analytial solutions of the boundary layer problem for different parameters γ.

Hence, the problem we would like to solve is

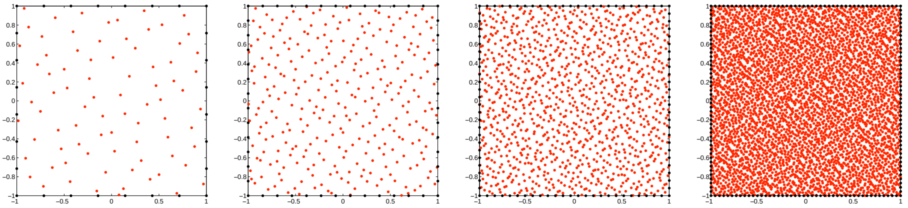
$$\Delta\theta(\mathbf{x}) = -q(\mathbf{x}) \quad \text{in } \Omega := [-1,1]^2 \tag{6.46}$$

with $\mathbf{x} \in \mathbb{R}^2$ and the source term defined in (6.44). Being an academic example, it is well-suited for validation of a numerical method, since the analytical solution is given by (6.45).

We apply the following boundary conditions

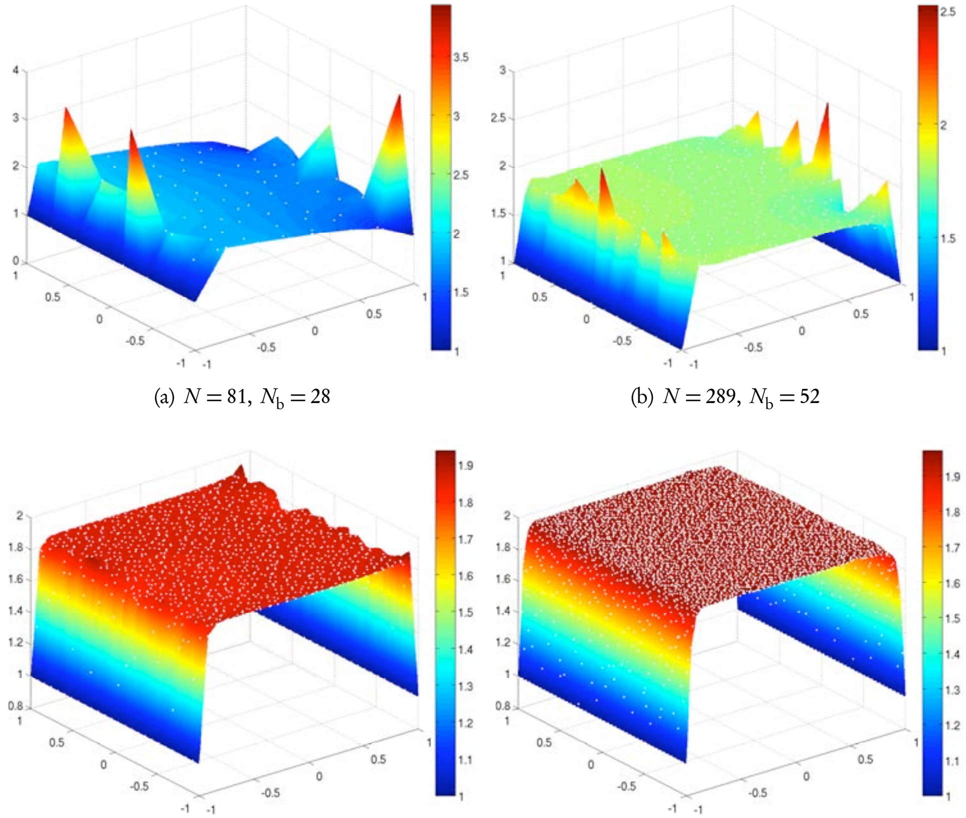
$$\begin{aligned} \theta(x=-1) &= \theta(x=1) = 1, & (6.47)\\ \theta(y=-1) &= \theta(y=1), & (6.48)\\ \frac{\partial}{\partial y}\theta(y=1) &= 0, & (6.49) \end{aligned}$$

where equation (6.47) is the Dirichlet boundary condition on two domain boundaries parallel to y-axis, and equations (6.48)–(6.49) define the periodic boundary conditions in y-direction. In order to study the accuracy of the method and to test the method for thin boundary layers with steep gradients we choose the case $\gamma = 30$. For the discretization of the problem (6.46) with boundary conditions (6.49) and in order to study convergence behaviour of the solution a series of sets of N Halton points within the domain augmented with N_b points uniformly distributed on the boundary of the domain is used. Figure 6.19 shows the respective point distributions for the convergence study of the method and the corresponding solutions of the problem are shown in Figure 6.20 on the following page.



(a) $N_\Sigma = 109$ (b) $N_\Sigma = 341$ (c) $N_\Sigma = 1189$ (d) $N_\Sigma = 4405$

Figure 6.19 | Point distributions of N_Σ collocation points consisting of N Halton points augmented on a boundary with N_b boundary points for convergence study of a solution of thin boundary layer problem.



(a) $N = 81,\ N_b = 28$ (b) $N = 289,\ N_b = 52$

(c) $N = 1089,\ N_b = 100$ (d) $N = 4225,\ N_b = 180$

Figure 6.20 | Convergence of the solution of a thin boundary layer problem with steep gradients computed on N Halton points augmented on a boundary with N_b boundary points.

Table 6.5 and Figure 6.21 show the convergence results of the method for the dilatation parameter $\rho = 2.5 \cdot h_{\Omega,\mathscr{S}}$, the exponent of the weight function $\alpha = 6$, and the quadratic basis (since the discretization of the Laplace operator requires second order derivatives). We obtained the expected third order of convergence for the RMS-error.

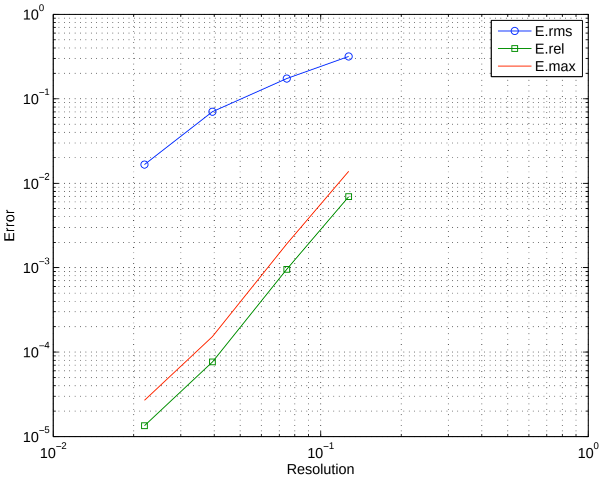


Figure 6.21 | Convergence plots of the errors for the thin boundary layer problem.

- ▶ For a reasonable range of STMCM parameters the results were of the same order. However, too high values of the dilatation parameter ρ or the exponent of the weight function α may result in a loss of order of convergence. In Section 6.5.6 on the next page the sensitivity of the STMCM to the choice of parameters will be studied.

Table 6.5 | Convergence study of thin boundary layer problem.

$h_{\Omega,\mathscr{S}}$	E_{rms}	E_{rel}	E_{max}	EOC_{rms}
1.2717e-01	3.2810e-01	8.0459e-03	1.6092e-02	–
7.4641e-02	1.6193e-01	1.2114e-03	2.4228e-03	1.3253
3.9390e-02	6.7912e-02	8.7456e-05	1.7491e-04	1.3595
2.1945e-02	1.1845e-02	2.0230e-05	4.0461e-05	2.9853

6.5.6 Helmholtz equation

Let us consider the 2d Helmholtz equation

$$\begin{aligned} \Delta u + \lambda^2 u &= f(x,y) \qquad x,y \in (-1,1), && (6.50)\\ u &= 0 \qquad \text{on the boundary} && (6.51) \end{aligned}$$

that arises in the analysis of wave propagation and various steady-state electromagnetic, acoustic, mechanical and thermal oscillation problems. For $\lambda = 0$ the Helmholtz equation reduces to the Laplace's, and with the right-hand-side zero – to the Poisson's equation. For a specific choice of $\lambda = 9$ and the forcing function

$$f(x,y) = \exp^{\left[-10\left((y-1)^2+(x-0.5)^2\right)\right]}$$

the solution of (6.50)–(6.51) is almost in resonance with the $(3,5)$-mode of an eigenfunction of the homogeneous Helmholtz equation, i.e. for $f(x,y)=0$, with the eigenvalue

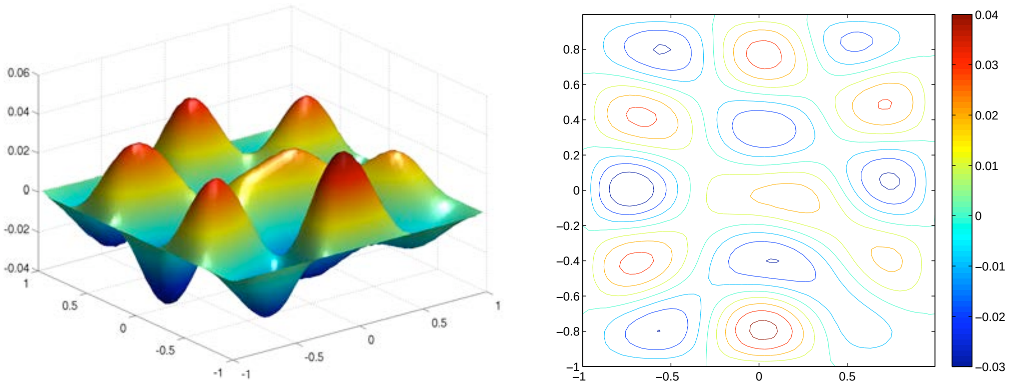
$$\lambda_{3,5} = \frac{1}{2}\sqrt{\pi^2(3^2+5^2)} = 9.15923782 \approx 9,$$

and, therefore, has roughly the form of a wave with five half-wavelengths in y- and three half-wavelengths in x-direction.

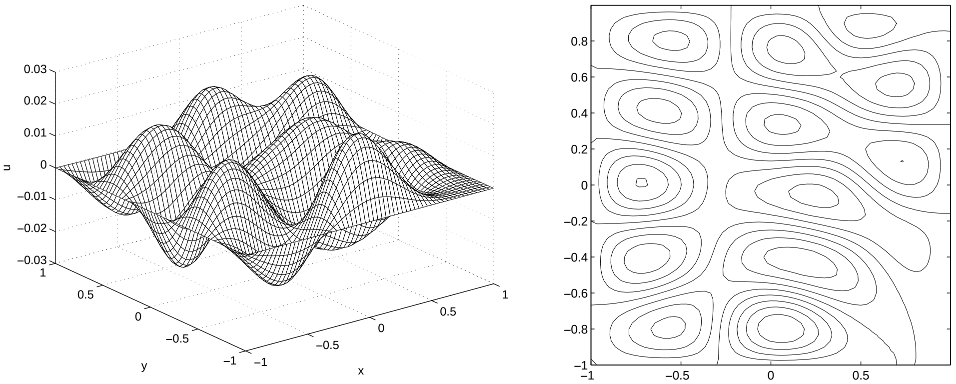
We compare our solution by STMCM with the Chebyshev pseudospectral solution [193] and solve the problem on a set of N Halton points augmented with a set of points uniformly distributed on a boundary. Figure 6.22 shows good agreement of these two solutions, and Figure 6.23 outlines the convergence behaviour of the STMCM solution. Table 6.6 contains the error norms as well as the experimental orders of convergence for one set of parameters.

Figure 6.24 and 6.25 demonstrate the sensitivity of the STMCM w.r.t. the IMLS parameters, namely the exponent of the weight function α and the size of support ρ (see Section 3.5.1 on page 43 for details) exemplarily for the absolute averaged error norms. One can clearly see that for a wide range of support sizes of the weight function the higher values of α lead to better results (Fig. 6.24). In this case the method is very robust for different values of ρ (Fig. 6.25). Moreover, for smaller supports the method is robust w.r.t. different values of α, but for larger ρ larger values of α should be chosen.

Choosing higher values of α seems to lead to better results. However, in this case the condition number of the matrix, which is to be inverted during the construction of the IMLS kernel functions, is getting worse very rapidly and the SVD-routine should be used with caution while finding the inverse, i.e. the consistency requirements on IMLS kernel functions should be checked for each computational point. If, in an unlikely case, the inversion fails, the appropriate adjustment of parameters should be performed.

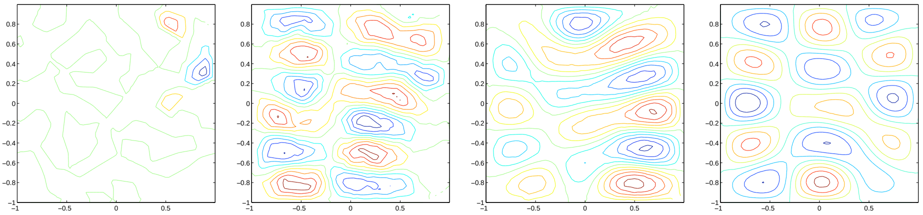


(a) *solution by STMCM*



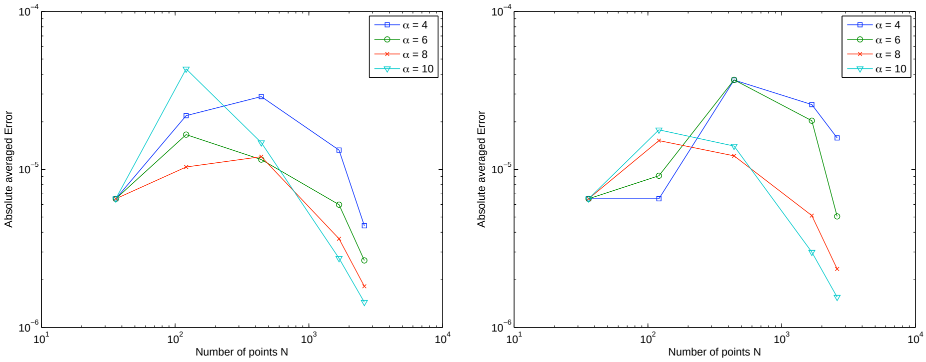
(b) *solution by Chebyshev pseudospectral method* [193]

Figure 6.22 | Solutions of the Helmholtz equtation (6.50)–(6.51) computed on a set of $N = 4225$ Halton collocation points augmented on the boundary.



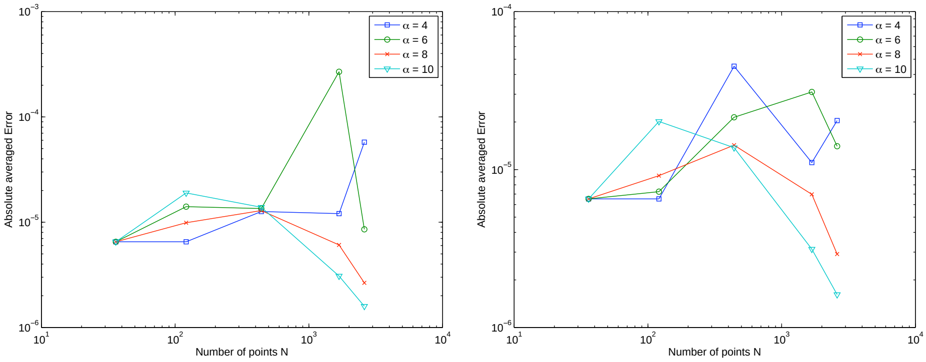
(a) $N = 81$ (b) $N = 289$ (c) $N = 1089$ (d) $N = 4225$

Figure 6.23 | Contour plots of the solution of the Helmholtz equation computed on a set of N Halton collocation points augmented on the boundary with other parameters kept constant.



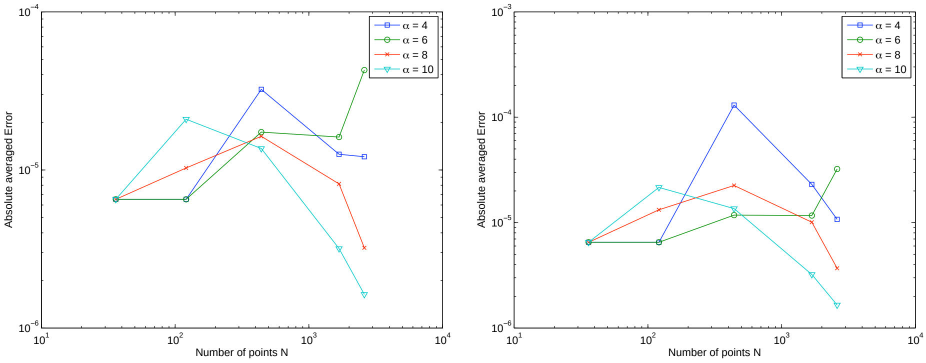
(a) $\rho = 2h_{\Omega,\mathscr{S}}$

(b) $\rho = 2.5h_{\Omega,\mathscr{S}}$



(c) $\rho = 3h_{\Omega,\mathscr{S}}$

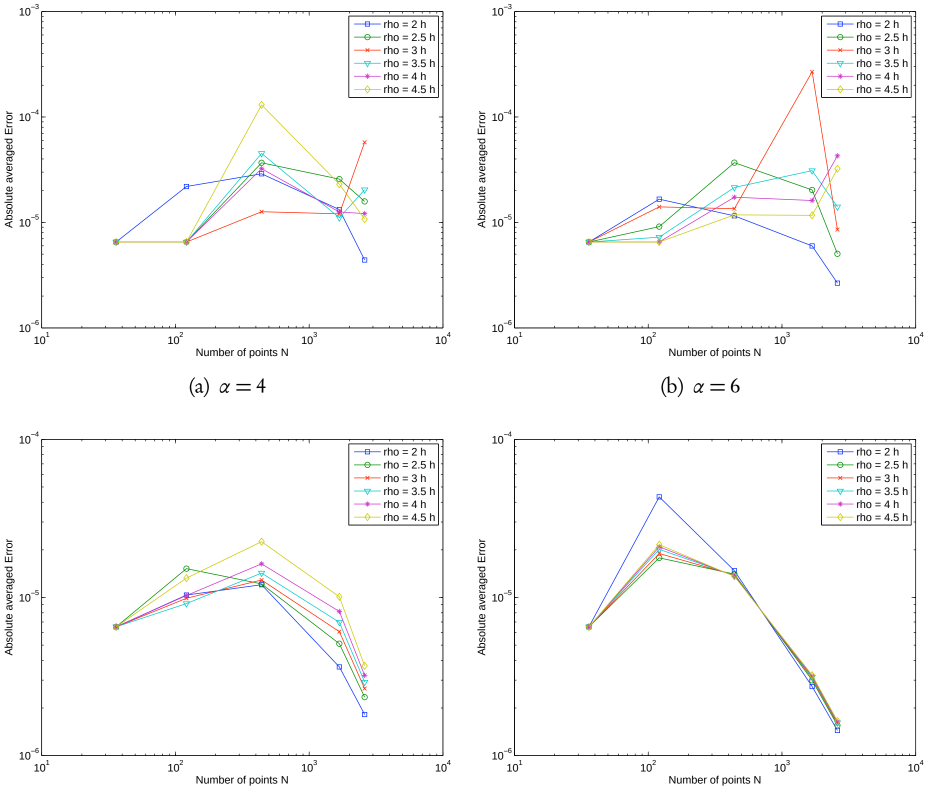
(d) $\rho = 3.5h_{\Omega,\mathscr{S}}$



(e) $\rho = 4h_{\Omega,\mathscr{S}}$

(f) $\rho = 4.5h_{\Omega,\mathscr{S}}$

Figure 6.24 | Parameter study of the convergence behaviour of the STMCM solution of Helmholtz equation for different supports of the weight function ρ.



(a) $\alpha = 4$

(b) $\alpha = 6$

(c) $\alpha = 8$

(d) $\alpha = 10$

Figure 6.25 | Parameter study of the convergence behaviour of the STMCM solution of Helmholtz equation for different exponents of the weight function α.

Table 6.6 | Convergence study of the Helmholtz equation ($\rho = 2.5 h_{\Omega,\mathscr{S}}$, $\alpha = 10$).

N_Σ	$\mathsf{E}_{\mathsf{rms}}$	$\mathsf{E}_{\mathsf{rel}}$	$\mathsf{E}_{\mathsf{max}}$	$\mathsf{EOC}_{\mathsf{rms}}$	$\mathsf{EOC}_{\mathsf{max}}$	$\mathsf{EOC}_{\mathsf{rel}}$
36	8.502484e-03	2.687455e-04	6.519398e-06	–	–	–
121	1.686321e-02	1.780929e-05	7.341425e-04	-0.9879	-1.4498	-1.4498
441	1.758915e-02	1.405531e-05	5.793942e-04	-0.0608	0.3415	0.3415
1681	4.204464e-03	2.995771e-06	1.234930e-04	2.0647	2.2301	2.2301
2601	2.185695e-03	1.554748e-06	6.409050e-05	2.9318	2.9393	2.9393

6.6 Time-Dependent Partial Differential Equations

Many time-dependent problems in engineering are described by parabolic partial diffrential equations (see Section 6.2.1 on page 74), i.e. the problems with one marching direction, satisfying equilibrium conditions in other directions. This kind of equations appear in heat and mass transfer problems, in combustion theory, biology, ecology, crystallization, plasma physics, etc., and the corresponding physical processes are convection and diffusion. On the other hand, a frequent phenomenon of a wave appears in natural applications, e.g. electromagnetic waves, acoustic waves, surface waves, vibrating strings and membranes, etc. These phenomena are described by hyperbolic partial differential equations.

The major difference between diffusions and waves is the character of transportation of information. Whereas in waves information and initial singularities are transported throughout domain with a certain finite propagation speed, in diffusion processes they are available immediately in the whole domain and eventually disappear. Moreover, waves are reversible from mathematical point of view, i.e. the problem is well-posed for negative time instants $t < 0$, which is not the case for diffusion processes – the temperature of a body or the original concentration of a diffusing gas or liquid cannot be determined for previous time instants, provided only the current state is available.

6.6.1 Diffusions :: Heat equation

Let us consider the following diffusion problem on a disc $\Omega : x^2 + y^2 < R^2$ with boundary $\Gamma : x^2 + y^2 = R^2$ in the time interval $I = [0, T]$

$$\begin{aligned} \frac{\partial u}{\partial t} &= \varkappa \Delta u \quad \text{in} \quad \Omega, && (6.52) \\ u(x,y,t) &= 0 \quad \text{on} \quad \Gamma, && (6.53) \end{aligned}$$

and initial condition

$$u(x,y,t=0) = f(\sqrt{x^2+y^2}), \tag{6.54}$$

where $\varkappa$ is called thermal conductivity in the context of heat transfer.

For initial data of chosen form (6.54) the solution is radially symmetric and using the transformation of the Laplacian into polar coordinates r and θ (see Appendix B on page 155) the equations (6.52)–(6.54) take form

$$\begin{aligned} \frac{\partial u}{\partial t} &= \varkappa \left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} \right) \quad \text{in} \quad Q = \Omega \times I, \\ u(r,t) &= 0 \quad \text{on} \quad \Gamma, \\ u(r,t=0) &= f(r). \end{aligned} \tag{6.55}$$

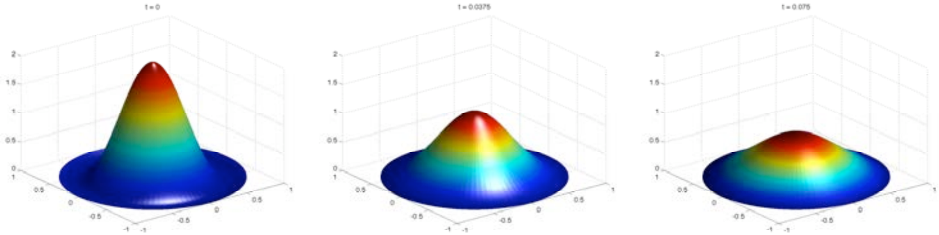
For $\varkappa = 1$ and with specific choice of initial conditions

$$f(r) = \sum_{i=1}^{2} \mathrm{J}_0(\zeta_i r), \tag{6.56}$$

where J_0 is the Bessel function of the first kind of order zero (see Appendix A on page 151) and $\zeta_1 = 2.4048$, $\zeta_2 = 5.5201$ are its first two roots [3], the analytical solution is known

$$u^{\mathrm{ex}}(r,t) = \sum_{i=1}^{2} \mathrm{J}_0(\zeta_i r)\,\mathrm{e}^{-\zeta_i^2 t}, \tag{6.57}$$

and hence we can study the accuracy and convergence of the STMCM. Figure 6.26 shows the STMCM solutions of (6.55) with $\varkappa = 1$ and initial condition (6.56).



(a) *initial condition*, $t = 0$ (b) $t = 0.0375$ (c) $t = 0.075$

Figure 6.26 | Solution of the heat equation (6.55) on a disc by STMCM for $\varkappa = 1$ and initial condition (6.56).

Figure 6.27 and Table 6.7 show the convergence behaviour of the STMCM solution of the heat equation on a unit circle. For this example and chosen IMLS-parameters ($\alpha = 6$, $\rho = 2.5h_{\Omega,\mathscr{S}}$) and space-time-parameters ($\delta_n = 2\Delta t$) we obtained the second order of convergence for the root-mean-square error and the fourth order of convergence for the averaged relative and absolute errors.

Table 6.7 | Convergence results for the heat equation (6.55).

δ_n	$\mathrm{E_{rms}}$	$\mathrm{E_{abs}}$	$\mathrm{E_{rel}}$	$\mathrm{EOC_{rms}}$	$\mathrm{EOC_{abs}}$	$\mathrm{EOC_{rel}}$
0.2	5.4042e-02	8.0983e-03	4.0492e-03	–	–	–
0.1	3.1156e-02	2.6020e-03	1.3010e-03	0.7946	1.6380	1.6380
5.00e-02	1.7229e-02	4.8256e-04	2.4128e-04	0.8547	2.4308	2.4308
2.50e-02	4.6447e-03	5.4262e-05	2.7131e-05	1.8911	3.1527	3.1527
1.25e-02	9.9645e-04	4.0523e-06	2.0262e-06	2.2207	3.7431	3.7431
6.25e-03	2.1665e-04	2.4602e-07	1.2301e-07	2.2014	4.0419	4.0419

► Computational errors were computed over the whole space-time domain Q according formulae given in Section 5.1 on page 63. Here as well as in the following examples of this section it is convenient to use for convergence studies a regular grid with equal

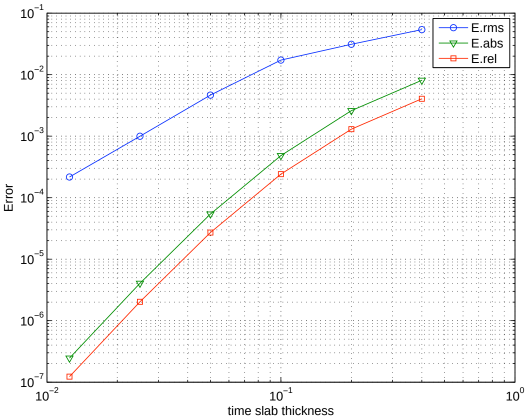


Figure 6.27 | Convergence plot of the STMCM solution of the heat equation (6.55).

spacings Δt and Δx in time and space, respectively. Unless stated differently, each of the space-time slabs is discretized by three points in time direction, i.e. $\delta_n = 2\Delta t$.

6.6.2 Vibrations :: Wave equation

In this section we will consider PDEs of hyperbolic type that often require more care when treated numerically, due to the fact that spurious oscillations may appear and, even worse, be amplified ruining herewith the numerical solution.

String vibrations

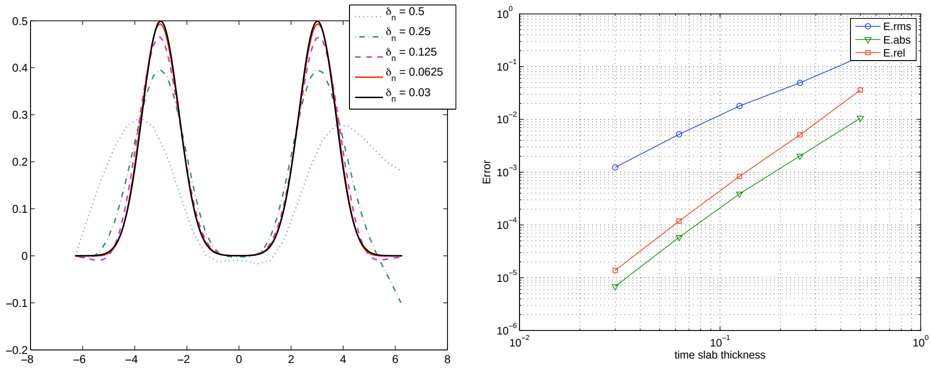
Let us consider a wave equation in a space-time domain $Q = \Omega \times [0, T]$

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \quad \text{in } Q \tag{6.58}$$

with initial and boundary conditions

$$\begin{aligned} u(x, t=0) &= u_0(x), \\ \frac{\partial u}{\partial t}(x, t=0) &= v_0(x) \\ u(x,t) &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{6.59}$$

which describes vibration of a string.



(a) *solutions at $t = T$*

(b) *convergence plots*

Figure 6.28 | Solutions of the wave equation (6.58) by STMCM at the end of time interval $t = T = 1.5$ for different δ_n with a constant speed $c = 2$ and convergence plots of the errors.

It is well known that for such an initial boundary value problem the exact solution is given by the d'Alembert formula

$$u^{\text{ex}}(x,t) = \frac{1}{2}\left[u_0(x+ct) + u_0(x-ct)\right] + \frac{1}{2c}\int_{x-ct}^{x+ct} v_0 \, d\tau. \tag{6.60}$$

As can be seen from (6.60) the initial wave is split into two travelling in opposite directions with the speed c, and the integral term depends on the initial velocity v_0.

- A difference to the diffusion equation must be noted. Whereas in the previous example only the displacement initial condition is required (i.e. the condition of Dirichlet type), in the case of waves one also needs initial values for velocity (i.e. also the condition of Neumann type). This is accomplished by incorporating velocity degrees of freedom into the vector of unknowns.

We solve the equation (6.58) with initial conditions $u_0 = \mathrm{e}^{-x^2}$ and $v_0 = 0$ by STMCM, the solution is shown for parameters $\alpha = 6, \rho = 2.5h_{\Omega,\mathscr{S}}$, quadratic basis, $\delta_n = 2\Delta t$ in Figure 6.29. For this specific choice the exact solution is

$$u^{\text{ex}}(x,t) = \frac{1}{2}\left[\mathrm{e}^{-(x+ct)^2} + \mathrm{e}^{-(x-ct)^2}\right], \tag{6.61}$$

and we may compute the error between the STMCM solution and the exact one. Figure 6.28 shows the STMCM solution of the wave equation at the end of time interval $t = T$ for different thicknesses of each space-time slab δ_n.

Table 6.8 | Convergence results for the wave equation (6.58).

δ_n	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
0.5	1.5267e-01	1.0584e-02	3.5939e-02	–	–	–
0.25	4.9393e-02	2.0222e-03	5.1013e-03	1.6281	2.3879	2.8166
0.125	1.7985e-02	3.8855e-04	8.3493e-04	1.4575	2.3798	2.6111
0.0625	5.2268e-03	5.8422e-05	1.1842e-04	1.7828	2.7335	2.8177
0.03	1.2322e-03	6.8585e-06	1.3738e-05	1.9688	2.9187	2.9349

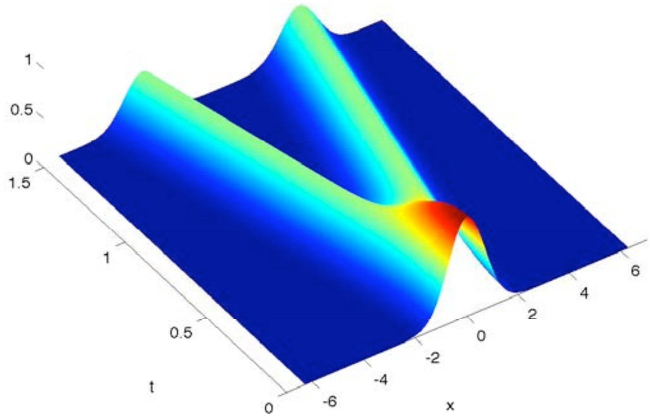


Figure 6.29 | STMCM solution of the wave equation (6.58).

As one would expect, decreasing values of δ_n lead to smaller errors between the computational solution and the exact one, see Figure 6.29, we obtain third order convergence for absolute and relative averaged errors and second order – for the RMS-error (see Table 6.8 on the facing page).

- ▶ In this example we computed the error between the known analytical solution and the numerical one at the end of the time interval, i.e. $t = T$. The computation of errors over the whole domain Q is also possible, but that does not change the order of convergence.

As a parameter study has shown, the STMCM is rather robust since there is only a slight influence of different values of ρ and α on the convergence behaviour of the numerical solution if the time slab thickness δ_n is kept constant (here $\delta_n = 0.01$), see Figures 6.30 and 6.31. The same is valid for different values of the `CFL` number defined as a ratio of time and space discretizations, i.e. `CFL` $=\Delta t/\Delta x$, see Figure 6.32.

Membrane vibrations

Let us now study a 2d counterpart to the 1d string vibrations discussed in the previous section, namely the vibration of membranes. An analogue to the heat equation considered in Section 6.6.1 on page 110 is the following wave equation on a circle describing vibrations of a circular membrane fixed on a frame.

Consider the following wave problem on a disc $\Omega : x^2+y^2 < R^2$ with boundary $\Gamma : x^2+y^2 = R^2$ in the time interval $I = [0, T]$

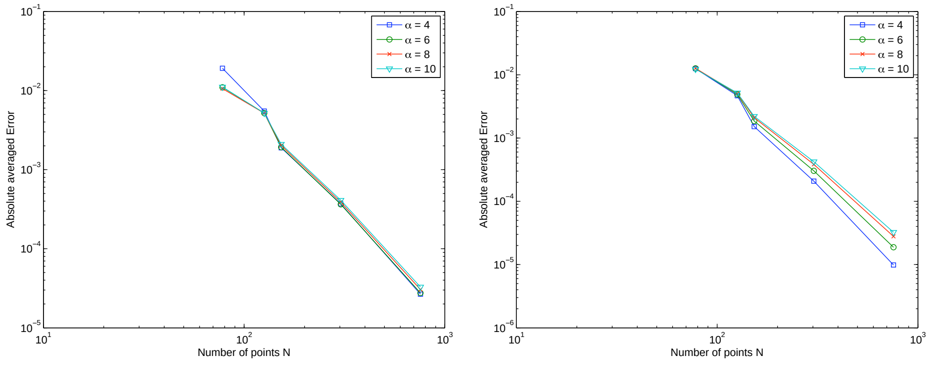
$$\begin{aligned} \frac{\partial^2 u}{\partial t^2} &= c^2 \Delta u \quad \text{in} \quad \Omega, && (6.62)\\ u(x,y,t) &= 0 \qquad \text{on} \quad \Gamma, && (6.63) \end{aligned}$$

and initial conditions

$$\begin{aligned} u(x,y,t=0) &= f(\sqrt{x^2+y^2}), && (6.64)\\ \frac{\partial u}{\partial t}(x,y,t=0) &= g(\sqrt{x^2+y^2}), && (6.65) \end{aligned}$$

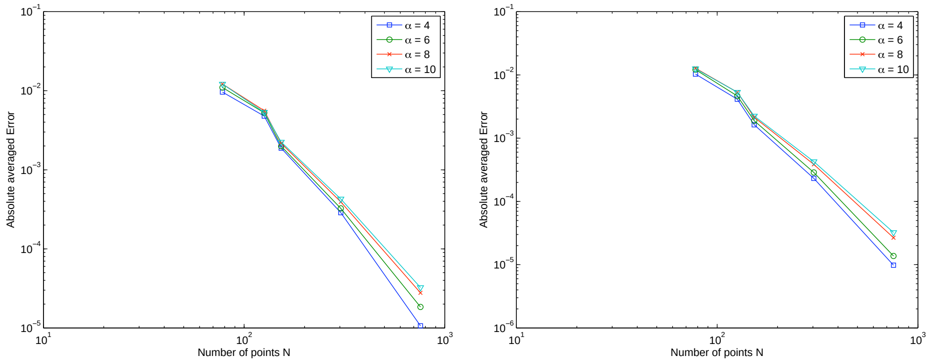
where c is the wave speed. For a specific choice of initial data

$$\begin{aligned} f(r) &= \sum_{i=1}^{2} \mathrm{J}_0(\zeta_i r), && (6.66)\\ g(r) &= 0, && (6.67) \end{aligned}$$



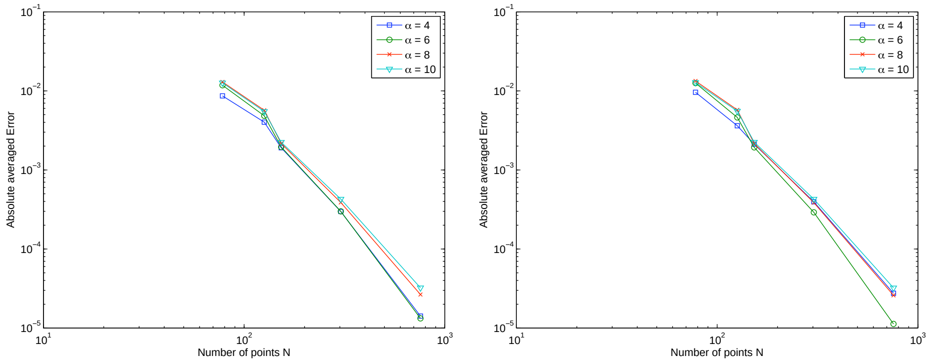
(a) $\rho = 2h_{\Omega,\mathscr{S}}$

(b) $\rho = 2.5h_{\Omega,\mathscr{S}}$



(c) $\rho = 3h_{\Omega,\mathscr{S}}$

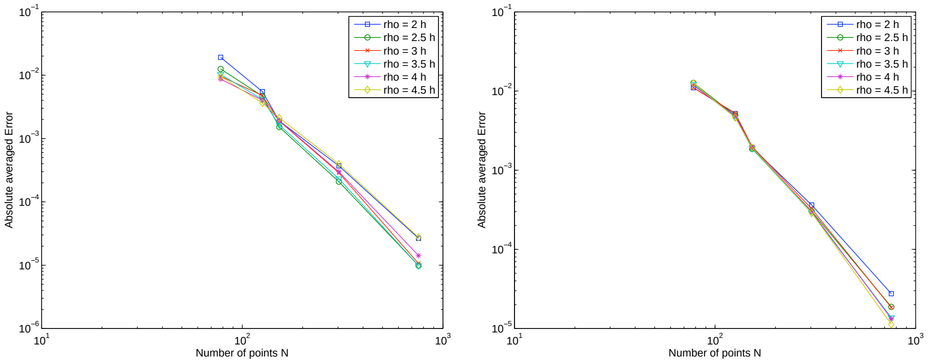
(d) $\rho = 3.5h_{\Omega,\mathscr{S}}$



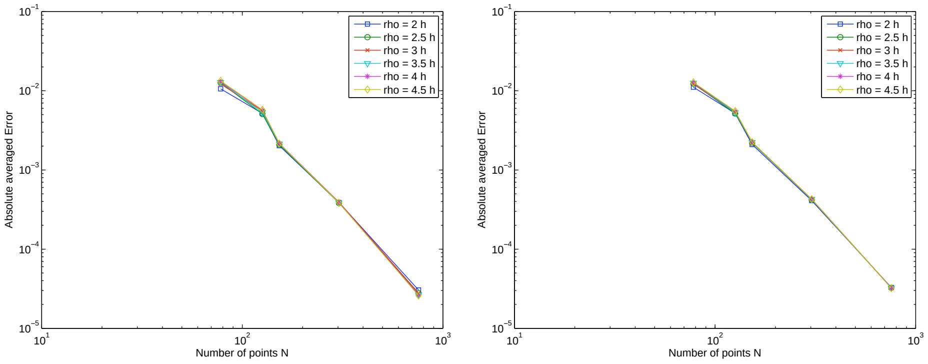
(e) $\rho = 4h_{\Omega,\mathscr{S}}$

(f) $\rho = 4.5h_{\Omega,\mathscr{S}}$

Figure 6.30 | Parameter study of the convergence behaviour of the STMCM solution of wave equation (6.58) for different supports of the weight function ρ.



(a) $\alpha = 4$ (b) $\alpha = 6$



(c) $\alpha = 8$ (d) $\alpha = 10$

Figure 6.31 | Parameter study of the convergence behaviour of the STMCM solution of wave equation (6.58) for different exponents of the weight function α.

where J_0 is the Bessel function of the first kind of order zero (see Appendix A on page 151) and $\zeta_1 = 2.4048$, $\zeta_2 = 5.5201$ are its first two roots [3]. The exact solution to this problem is radially symmetric and reads as

$$u^{\mathrm{ex}}(x,y,t) = u(r,t) = \sum_{i=1}^{2} J_0(\zeta_i r)\cos(\zeta_i t). \tag{6.68}$$

Using the transformation of the Laplacian into polar coordinates (see Appendix B on

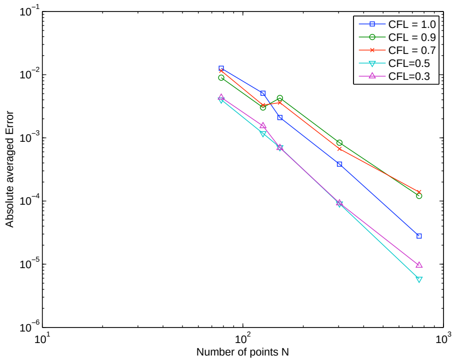


Figure 6.32 | Influence of the CFL number on the STMCM solution of wave equation (6.58) for $\rho = 2.5h_{\Omega,\mathscr{S}}$, $\alpha = 8$.

page 155) the equations (6.62)–(6.64) take the form

$$\begin{aligned}
\frac{\partial^2 u}{\partial t^2} &= c^2\left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r}\right) \quad \text{in} \quad Q = \Omega \times I, \\
u(r,t) &= 0 \quad \text{on} \quad \Gamma, \\
u(r,t=0) &= f(r), \quad \frac{\partial^2 u}{\partial t^2}(r,t=0) = g(r),
\end{aligned} \tag{6.69}$$

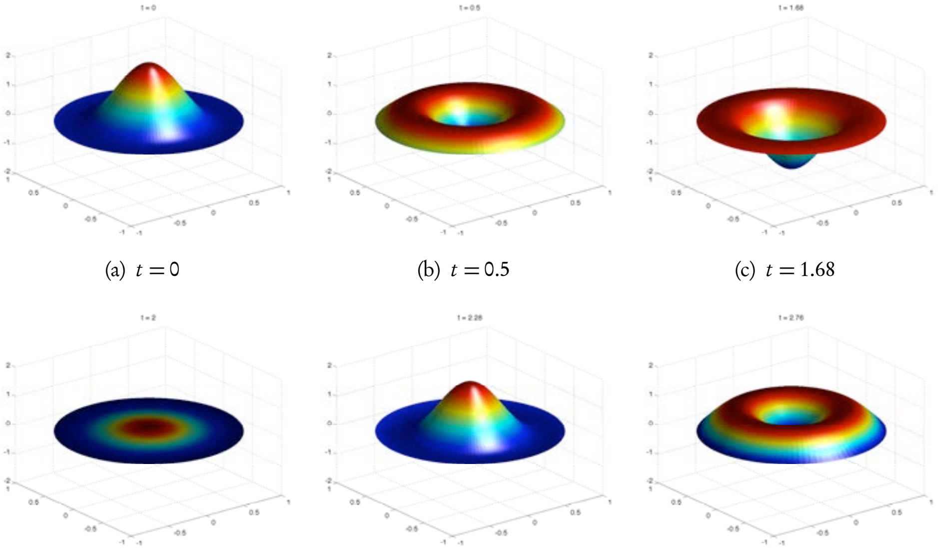
where $f(r)$ and $g(r)$ are given in (6.66)–(6.67). Figure 6.33 on the facing page shows the STMCM solution of (6.69) for different time instants computed for the following set of parameters: $\alpha = 6$, $\rho = 2.5h_{\Omega,\mathscr{S}}$, quadratic basis, $\delta_n = 0.01$.

Table 6.9 on the next page demonstrates the convergence behaviour of the STMCM applied to the wave equation of a vibrating circular membrane (6.69). We obtain the second order convergence for the RMS-error and the fourth order for the averaged absolute and relative errors.

- ▶ Computational errors were computed over the whole space-time domain Q according formulae given in Section 5.1 on page 63.

6.6.3 Convection :: Transport equation

In this section we will investigate the accuracy and convergence of the STMCM when applied to a simple convection problem with a constant and variable speed, which describes e.g. the transport of a contaminant in a tube with a flow of an appropriate speed.



(a) $t = 0$ (b) $t = 0.5$ (c) $t = 1.68$

(d) $t = 2$ (e) $t = 2.28$ (f) $t = 2.76$

Figure 6.33 | STMCM solution of the wave equation (6.69).

Table 6.9 | Convergence results for the wave equation (6.69).

δ_n	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
2.00e-01	2.6487e-01	9.6267e-03	4.8134e-03	–	–	–
1.00e-01	9.3958e-02	1.1164e-03	5.5818e-04	1.4952	3.1082	3.1082
5.00e-02	2.2072e-02	7.4859e-05	3.7430e-05	2.0898	3.8985	3.8985
2.50e-02	4.9883e-03	4.6594e-06	2.3297e-06	2.1456	4.0060	4.0060
1.25e-02	1.1754e-03	2.8892e-07	1.4446e-07	2.0853	4.0114	4.0114
6.25e-03	2.7535e-04	1.6947e-08	8.4736e-09	2.0939	4.0915	4.0915

A transport equation in a space-time domain $Q = \Omega \times I : [0, L] \times [0, T]$ reads as

$$\frac{\partial u}{\partial t} + c(x)\frac{\partial u}{\partial x} = 0 \quad \text{in } Q \tag{6.70}$$

with initial condition

$$u(x, t = 0) = u_0 = \mathrm{e}^{-10(x-1)^2}. \tag{6.71}$$

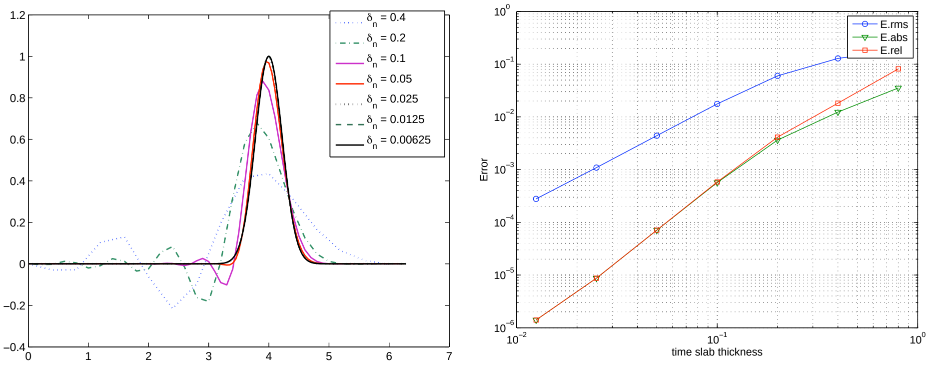
We solve the equation (6.70)–(6.71) on a domain $\Omega = [\,0, 2\pi]$ in time interval $[0, 6]$ and consider two cases of the speed function: (i) a constant speed $c(x) = c = 0.2$, (ii) a variable speed function $c(x) = 0.2 + \sin^2(x)$. As in previous examples, the whole space-time domain is divided into space-time slabs of thickness δ_n and the solution is sequentially found for each of the slabs. The solution at the end of each of the space-time slabs is used as initial solution for the next slab.

In order to study the convergence properties it is convenient to choose a regular grid with equal spacings Δt and Δx in time and space direction, respectively, as well as the constant thickness of each of the space-time slabs $\delta_n = 2\Delta t$ (the minimum time slab thickness required for the used quadratic basis function). The number of space-time slabs needed to compute the solution for the whole time interval changes depending on the size of δ_n. Table 6.10 and Figure 6.34 show the convergence results. We obtain the expected third order of convergence for this equation.

Table 6.10 | Convergence results for the linear transport equation (6.70) with a constant speed.

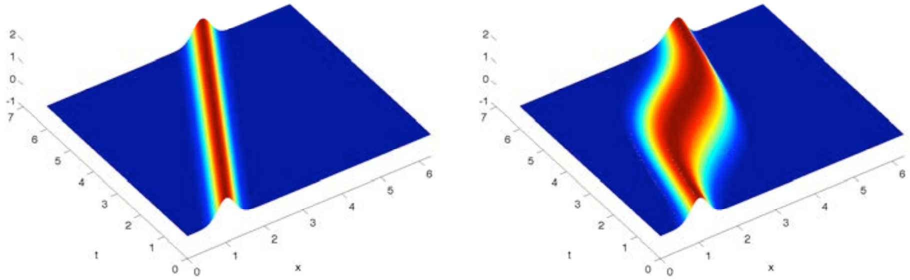
δ_n	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
0.4	1.8287e-01	3.5335e-02	8.1295e-02	–	–	–
0.2	1.2871e-01	1.2347e-02	1.8176e-02	0.5067	1.5170	2.1611
0.1	6.0085e-02	3.6314e-03	4.1266e-03	1.0990	1.7655	2.1390
0.05	1.7616e-02	5.6776e-04	5.8289e-04	1.7701	2.6772	2.8237
0.025	4.3994e-03	7.0898e-05	7.1152e-05	2.0015	3.0015	3.0342
0.0125	1.0915e-03	8.7262e-06	8.7295e-06	2.0109	3.0223	3.0269
0.00625	2.7819e-04	1.4021e-06	1.4020e-06	1.9722	2.6377	2.6384

For smaller time slab thicknesses the observed numerical oscillations diminish and the numerical solution approaches the exact one in good agreement as shown in Figure 6.34. Solutions of a linear transport equation in the whole space-time domain Q for the constant and variable speed functions computed with $\delta_n = 0.0125$ successively for 240 space-time slabs are shown in Figure 6.35 on the next page.



(a) *solutions at $t = T$* (b) *convergence plots*

Figure 6.34 | Solutions of the transport equation (6.70) by STMCM at the end of time interval $t = T = 6$ for different δ_n with a constant speed $c = 0.2$ and convergence plots of the errors.



(a) $c(x) = c = 0.2$ (b) $c(x) = 0.2 + \sin^2(x)$

Figure 6.35 | Solutions of the transport equation (6.70) by STMCM for different speeds $c(x)$.

6.6.4 Hamilton-Jacobi equations :: Level-set equation

Hamilton-Jacobi equations are hyperbolic PDEs of the form

$$\frac{\partial \phi}{\partial t} + H[\mathbf{x}, \nabla \phi] = 0, \tag{6.72}$$

where H is a Hamiltonian operator. These equations are often used to describe moving interfaces and to some extent they are similar to the Schrödinger equation of relativistic quantum mechanics.

A special case of Hamilton-Jacobi equations is the level-set equation (in combustion theory the level-set equation is known as the G-equation that describes the evolution of a flame front) [162, 161] which describes the movement of an implicit surface ϕ in an extraneous velocity field $\mathbf{v}$

$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0 \qquad \text{in} \quad Q, \tag{6.73}$$

with initial condition

$$\phi_0 = \phi(\mathbf{x}, t = 0) \qquad \text{in} \quad \Omega. \tag{6.74}$$

- ▶ In 1d the level-set equation can be written in an equivalent form of conservation laws. In this case deriving the level-set equation

  $$\frac{\partial \phi}{\partial t} + v \cdot \frac{\partial \phi}{\partial x} = 0$$

  w.r.t x, we obtain

  $$\frac{\partial}{\partial x}\left[\frac{\partial \phi}{\partial t}\right] + \frac{\partial}{\partial x}\left[v \cdot \frac{\partial \phi}{\partial x}\right] = 0,$$

  which can be written as

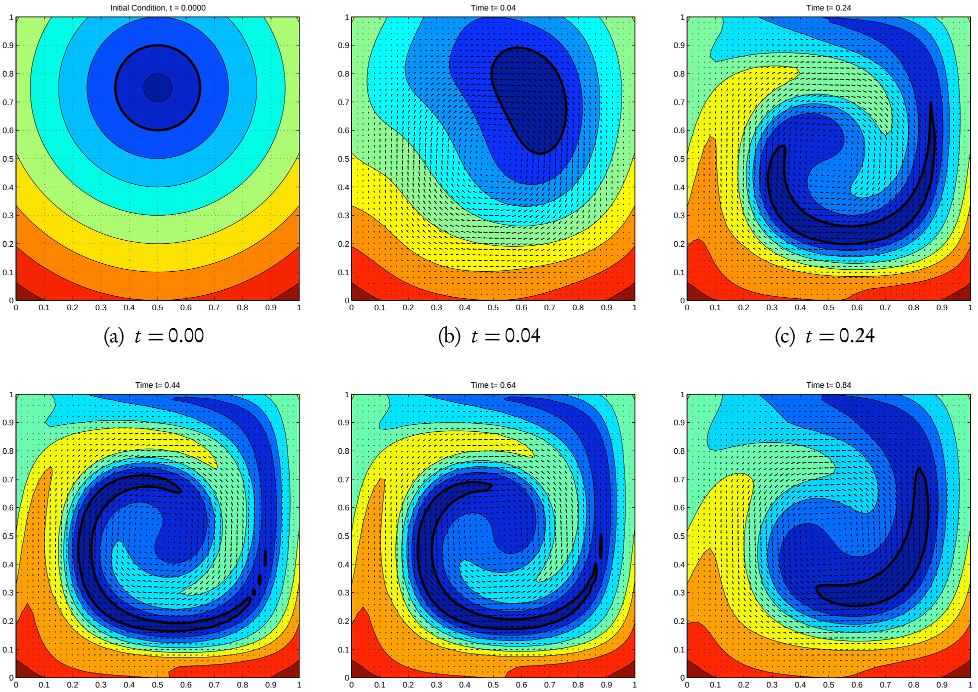
  $$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}[vU] = 0.$$

  The last equation is a conservation law for a scalar quantity $U := \frac{\partial \phi}{\partial x}$.

In Chapter 7 on page 131 the level-set equation will be used to describe the moving boundaries in a growing biofilm. Therefore, in this section we must check the accuracy of the STMCM on this kind of equations. Hereby the convergence of the method in time direction is of major importance.

Using the presented STMCM we solve the level-set equation (6.73) on a domain $Q = \Omega \times I = ([0,1] \times [0,1]) \times [0,1]$ with the initial condition (6.74) defined as a distance function

$$\phi_0 = \sqrt{(x_1 - x_{1c})^2 + (x_2 - x_{2c})^2} - R.$$



(a) $t = 0.00$ (b) $t = 0.04$ (c) $t = 0.24$

(d) $t = 0.44$ (e) $t = 0.64$ (f) $t = 0.84$

Figure 6.36 | Solution of the level-set equation with a deformative velocity field.

The initial zero level of the level set function ϕ forms a circle with center coordinates $x_{1c} = 0.5$, $y_{2c} = 0.75$, and the radius $R = 0.15$.

The velocity field is defined as

$$\mathbf{v} = \begin{pmatrix} -\cos \pi t \cdot \left[8 \sin^2 \pi x_1 \sin \pi x_2 \cos \pi x_2 \right] \\ \cos \pi t \cdot \left[8 \sin^2 \pi x_2 \sin \pi x_1 \cos \pi x_1 \right] \end{pmatrix}. \tag{6.75}$$

Figure 6.36 shows the level-set function and the velocity field defined in (6.75) for different moments in time. The velocity field is constructed in a way that it acts clockwise for $0 < t < 0.5$, and counterclockwise for $0.5 \leqslant t < 1$, so that the level-set function comes to its initial position for $t = 1$. The zero level of the level-set function is shown with a black solid line. For this example we chose even space discretization steps $\Delta x_1 = \Delta x_2 = \Delta x = 0.02$. The time slab thickness was chosen as $\delta_n = 2\Delta t = 0.04$.

We measure the errors between the initial condition at $t = 0$ and the solution at $t = 1$ that must be identical to the initial condition by construction due to a specific choice of the deformative velocity field. The space discretization is kept constant at $\Delta x = 5.2632\text{e-}02$, and the time discretization step Δt is decreased. The measured error norms give information on both accuracy and convergence behaviour of the method. Table 6.11 gives an overview of the error norms and the order of convergence in time for STMCM parameters $\rho = 2.5 h_{\Omega,\mathcal{S}}$ and $\alpha = 6$. We obtain the fourth order of convergence w.r.t. the infinity norm using the quadratic basis in time, and the third order – using the linear one. The convergence rates are higher than expected, since with decreasing time discretization the `CFL` -number is getting smaller too, which reduces the error.

Table 6.11 | Convergence behaviour of the solution of the level-set equation (6.73) with the deformative velocity field (6.75). Case (a): linear basis in space, quadratic in time. Case (b): linear basis in both space and time.

Δt	$\|\|u^{\text{ex}} - u^{\text{appx}}\|\|_2$	$\|\|u^{\text{ex}} - u^{\text{appx}}\|\|_\infty$	$\|\|u^{\text{ex}} - u^{\text{appx}}\|\|_2/\|\|u^{\text{ex}}\|\|_2$	EOC_∞
		Case (a)		
5.0000e-02	6.5609e-01	1.5553e-01	8.9978e-02	–
2.5000e-02	7.1343e-02	3.0224e-02	9.7841e-03	2.3634
1.2500e-02	5.0647e-03	2.3888e-03	6.9458e-04	3.6613
6.2500e-03	2.9407e-04	1.3766e-04	4.0329e-05	4.1171
3.1250e-03	1.9157e-05	8.6318e-06	2.6273e-06	3.9953
		Case (b)		
5.0000e-02	1.1673e+00	2.4773e-01	1.6008e-01	–
2.5000e-02	1.4625e-01	5.3292e-02	2.0056e-02	2.2168
1.2500e-02	3.5818e-03	1.5346e-03	4.9122e-04	5.1180
6.2500e-03	2.1686e-04	9.8784e-05	2.9740e-05	3.9574
3.1250e-03	2.9591e-05	1.2776e-05	4.0582e-06	2.9508

The results given in this section are presented by the author in [153].

6.6.5 Curvature driven evolving interfaces

A special case of the level-set equation – the curvature driven evolving interfaces – appears in a number of different applications: crystal growth, flame propagation, removing noise from images, interfaces between fluids, surface tension of bubbles and flexible membranes or when solving phase field equations, e.g. Cahn-Hilliard or Allen-Cahn equation. In this case the motion of the interface depends on geometric properties like curvature and normal direction only. Such problems are both exciting and ambitious and numerical solution with mesh-based established methods is a challenge. Moreover, as we shall see momentarily, such kind of motion resembles to a geometric version of a nonlinear heat equation.

This section demonstrates the ability of STMCM to deal with such kind of problems and we start with a theorem from differential geometry.

Theorem 6.7 *Any simple closed curve that moves due to its curvature gradually transforms to a circle, shrinks and eventually disappears.*

Proof See [74]. □

A unit *normal vector* at any point lying on a surface of a level-set function ϕ is defined as

$$\mathbf{n} = \frac{\nabla \phi}{|\nabla \phi|}, \tag{6.76}$$

and *curvature* κ is defined as the divergence of the normal vector, i.e.

$$\kappa = \nabla \cdot \mathbf{n} = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}. \tag{6.77}$$

In this section we are interested in surface movements due to curvature with normal velocity

$$\mathbf{v}_n := \mathbf{v} \cdot \mathbf{n} = -\kappa. \tag{6.78}$$

Inserting (6.78) into the level-set equation (6.73) we obtain

$$\frac{\partial \phi}{\partial t} = \left(\nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \right) |\nabla \phi| \tag{6.79}$$

which resembles to a nonlinear heat equation.

- If the level-set function is defined as a signed distance function, then $|\nabla \phi| = 1$ and the equation reduces to an unsteady heat equation with the thermal conductivity κ.
- In this section we are not interested in convergence properties of the STMCM when applied to the level-set equation, since this case is studied in Section 6.6.4. Here we only concentrate on a qualitative description of the solution behaviour. Motion due to curvature is of importance in many applications mentioned at the beginning of this section. This phenomenon, due to its diffusion-like character, may be used as additional viscosity term in mathematical models of physical phenomena, for representation of surface tension in flexible membranes or bubbles as well as for reshaping boundaries.

Figure 6.37 on the next page shows the STMCM solution of the level-set equation with a curvature driven velocity field. As initial condition we choose a parametric curve from Section 4.4 on page 54 for $\gamma = 3$. White areas represent the negative values level-set function and the moving interface is represented by a black line, in green areas the level-set function is positive.

The level-set equation (6.79) is discretized by STMCM using the thickness of each time slab $\delta_n = 0.005$ and $N = 882$ Halton points per time slab. It can be seen that the areas with negative curvature move quicker outwards and the shape gradually changes to a concave curve. Simultaneously, the curve shrinks and eventually disappears.

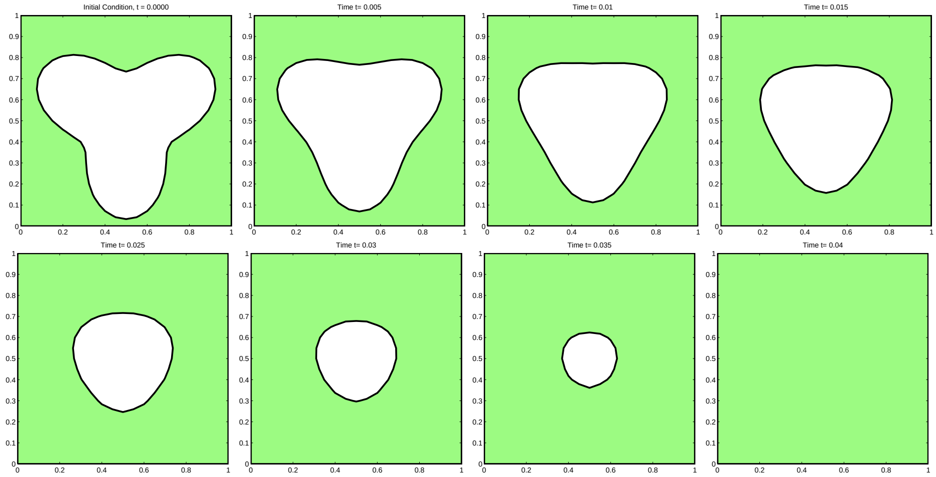


Figure 6.37 | STMCM solution of the level-set equation with a curvature driven velocity field.

6.7 Nonlinear Partial Differential Equations

As we have seen in Section 6.4 on page 89, a nonlinear deterministic system may result in an unpredictable behaviour even for a slight change of initial conditions. Moreover, the superposition principle fails in nonlinear systems and such phenomena as *shock waves* occur. As a result, nonlinear equations are much more difficult to treat numerically.

6.7.1 Shock waves :: Burgers' equation

Shock waves may develop in nonlinear problems even from homogeneous and smooth initial conditions. In this example we solve the nonlinear viscous Burgers' equation

$$\frac{\partial u}{\partial t}+\frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right)=\nu\frac{\partial^2 u}{\partial x^2}, \qquad 0\le x\le 1, \quad 0<t\le 1, \tag{6.80}$$

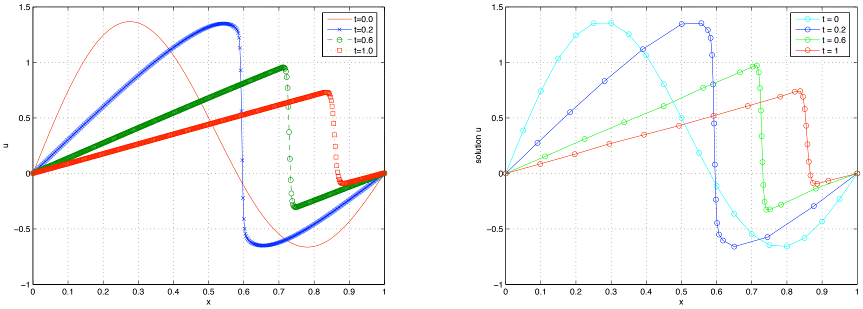
with the artificial viscosity $\nu = 2.0\text{e-}03$ and initial and boundary conditions

$$\begin{aligned} u(x,0) &= \sin(2\pi x)+0.5\sin(\pi x), && (6.81)\\ u(0,t) &= u(1,t)=0. && (6.82) \end{aligned}$$

The Burgers' equation is used to describe wave propagation in hydrodynamics and acoustics.

The nonlinearity is resolved using the fixed point iteration as described in Section 3.3 on page 38. The time slab thickness for this example $T_n = 0.04$, and the uniform point distribution with spacing $\Delta x = 2.22\text{e-}3$ is chosen.

As seen in Figure 6.38, the numerical solution of Burgers' equation (6.80)–(6.82) by STMCM is in excellent agreement with the reference solution obtained by a moving mesh method [92], where Burgers' equation is discretized with central differences and the moving mesh equation MMPDE6 is used. The sharp gradient develops as expected, and the solution gradually dissipates as $t \to \infty$.

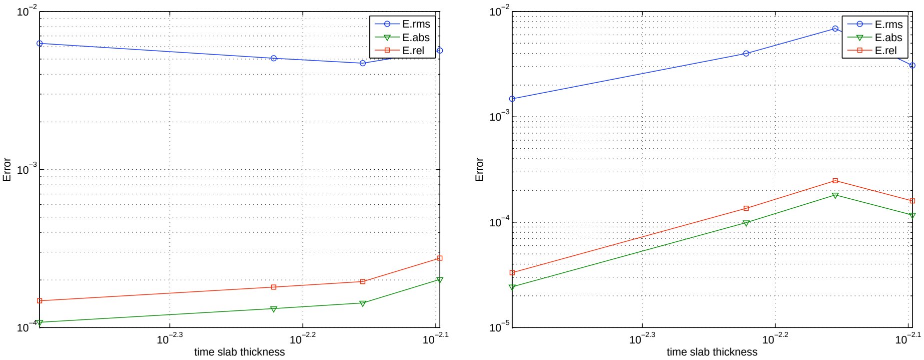


(a) *STMCM*

(b) *reference solution* [92]

Figure 6.38 | Solution of the nonlinear Burgers' equation by STMCM and a reference solution.

Using the solution by the moving mesh method computed for $N = 800$ points on an interval $[0, 1]$ as a reference, the convergence behaviour of the STMCM for different parameters of the weight function α is studied by measurements of errors between the two solutions at the end of the time interval at $t = 1$, see Figure 6.39 and Table 6.12.



(a) $\alpha = 8$

(b) $\alpha = 10$

Figure 6.39 | Convergence of the STMCM solution of the nonlinear Burgers' equation for different values of the parameter of the weight function α.

For $\alpha = 6$ the Picard iteration is unable to resolve the nonlinearity and hence there is no convergence. For $\alpha = 8$ the solution converges when the time slab thickness is reduced, yet the rate is very low. For $\alpha = 10$ a convergence rate of second order for the RMS-error is obtained.

Table 6.12 | Convergence behaviour of the STMCM solution of the nonlinear Burgers' equation.

δ_n	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
			$\alpha = 8$			
4.0000e-03	5.6591e-03	2.0192e-04	2.7518e-04	–	–	–
3.5000e-03	4.7051e-03	1.4325e-04	1.9559e-04	1.3827	2.5708	2.5567
3.0000e-03	5.0552e-03	1.3190e-04	1.8038e-04	-0.4656	0.5351	0.5251
2.0000e-03	6.2821e-03	1.0815e-04	1.4781e-04	-0.5359	0.4897	0.4912
			$\alpha = 10$			
4.0000e-03	3.0735e-03	1.1689e-04	1.5936e-04	–	–	–
3.5000e-03	6.8876e-03	1.8158e-04	2.4794e-04	-6.0427	-3.2989	-3.3102
3.0000e-03	3.9960e-03	9.9078e-05	1.3558e-04	3.5318	3.9300	3.9161
2.0000e-03	1.4824e-03	2.4372e-05	3.3214e-05	2.4457	3.4589	3.4690

- ▶ The case of an inviscid Burgers' equation, i.e. for $\nu = 0$, is even more challenging to be treated numerically. In this case some kind of a stabilizing diffusive term is needed; otherwise spurious oscillations are obtained and the numerical solution explodes.

6.7.2 Metastability :: Allen-Cahn equation

In this section we apply the STMCM to the bistable nonlinear reaction-diffusion Allen-Cahn equation

$$\frac{\partial u}{\partial t} = u - u^3 + \epsilon \frac{\partial^2 u}{\partial x^2}, \quad \epsilon > 0 \quad \text{const.} \tag{6.83}$$

This equation has three steady states: two stable ones $u = 1$, $u = -1$, and an unstable one $u = 0$. The numerical solution tends to develop flat areas near the stable steady states. One speaks in this case of a *metastability* effect, i.e. initial interfaces between the stable states may merge and the solution transforms to a state with only one interface.

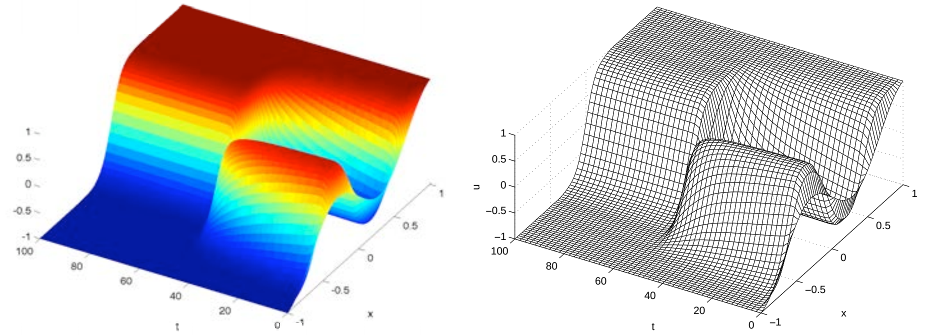
- ▶ Metastable phenomena are observed in the fields of thermodynamics, chemical systems, physics, nuclear physics, neuroscience, digital circuits, biology, etc.

Applying the STMCM to (6.83) with $\epsilon = 10^{-2}$ and initial condition

$$u(x, t = 0) = \frac{1}{2}\left[x + \sin\left(-\frac{3\pi}{2}x\right)\right] \tag{6.84}$$

and resolving nonlinearity according to strategy given in Section 3.3 on page 38, the STMCM solution is obtained. For parameters $\delta_n = 2\Delta t = 0.05$, $\alpha = 8$, and $\rho = 2.5h_{\Omega,\mathcal{S}}$

it is shown in Figure 6.40a. The STMCM solution is in excellent agreement with a reference solution obtained by a spectral Chebyshev method [193]. The initial condition (6.84) is metastable and as the flat regions near the two stable states $u = \pm 1$ increase, a rapid transition into a stable state is observed near $t = 55$.



(a) *STMCM* (b) *reference solution* [193]

Figure 6.40 | Solution of the nonlinear Allen-Cahn equation by STMCM and a reference solution.

6.8 Discussion

In this chapter we investigated the method proposed in this thesis applying it to a wide range of differential equations of various types. Not only the accuracy and convergence properties were studied, but also the sensitivity of the method to its parameters. In many cases we used known analytical solutions to measure numerical errors of computational results. In the cases, where no analytical solution is available, reference methods were used instead as a comparison.

Both for linear (a linear spring-mass oscillator, Foucault pendulum) and nonlinear (Lorenz attractors) ODEs the method gave good results. A comparative analysis of the onestep-like time integration scheme and the multistep-like one for a system of linear ODEs has shown that both approaches yield similarly accurate, stable, and convergent results. Yet for reasons of computational performance, it is advisable to subdivide the whole time interval into slabs and to compute the numerical solution sequentially for each of the time slabs. The STMCM is capable of solving nonlinear ODEs, which are highly sensitive to initial conditions, as it was demonstrated for Lorenz equations.

For stationary problems (elliptic PDEs) as well as for time-dependent ones (hyperbolic and parabolic PDEs) the method gave accurate and convergent results. The method is capable to solve diffusion, heat and wave propagation problems as well as convection and boundary layer problems accurately. The method also gives reliable results when applied to the level-set equation of Hamilton-Jacobi type which will be used to track the interface of moving domains in the following chapter.

For the nonlinear Burgers' equation it is well-known that the diffusive term acts as a stabilizing term. Yet also for this case shock waves can develop due to nonlinearity in the system even from initially smooth data that was also demonstrated by the STMCM solution. The inviscid case of the Burgers' equation is more challenging to be solved numerically and an additional stabilizing procedure is needed to avoid unphysical oscillations of the numerical solution. For the bistable nonlinear reaction-diffusion equation (Allen-Cahn equation) the STMCM solution is in an excellent agreement with the reference one.

"I do not think that the wireless waves I have discovered will have any practical application."

-- Heinrich Hertz

"What is of all things most yielding can overcome that which is the hardest."

-- Lao Tzu

CHAPTER 7

Application to Biofilm and Tumour Growth Modelling

In this chapter we apply the STMCM to coupled problems with evolving and deforming interfaces and changing topologies, namely the problem of biofilm growth and tumour growth. After a brief introduction a mathematical model of a biofilm as a continuum is presented. The model uses a two-biofluid representation governed by a system of incompressible Stokes equations and an implicit interface tracking technique via the level-set equation. Several validation examples are discussed. Finally, an existing mathematical model of tumour growth is considered and the accuracy of the numerical solution by STMCM is investigated.

7.1 Biofilms

In environments where moisture, nutrient, and a surface are present, one will most probably also find biofilms, which may consist of one or different species of bacteria, archaebacteria, algae as well as fungi or corrosion products. Biofilms can adhere to basically any type of surface (metal, plastics, medical implants, and tissue) by excreting a glue-like substance, the so-called EPS (extracellular polymeric substance) which protects the inner cells, assures the communication between the cells through bio-chemical signals and may also contain water channels for a nutrient and signal distribution.

Biofilms, as a natural part of the environment, have both harmful and positive impacts on it. Industry needs techniques to understand how they attach to the surface, colonize, grow and disperse. Much money is being spent nowadays due to effects caused by biofilms, e.g. equipment damage, energy losses, and product contamination. These affect the efficiency and safety of industrial processes.

But not only in the industry biofilms play a big role. According to [1] biofilms can be found in 80% of all infections, e.g. infections of the urinary tract, catheter, and middle-ear, formation of dental plaque or sinusitis. Hence, the biofilm research is also extremely important for human health. Since biofilms are a part of our natural environment, they can also be beneficial for treating drinking water, wastewater and for the detoxification of dangerous wastes.

In order to be able to get rid of the unwanted consequences of biofilms and control the beneficial biofilm development, it is important for engineers to have techniques at hand to model their behaviour.

7.2 State of research

This section will provide the reader with a brief overview of historical development as well as recent investigations in biofilm modelling.

All existing biofilm models can be divided into two big classes: discrete (lattice) models and continuum models. Lattice models are based on a probabilistic and discrete local rules. These models have some serious physical drawbacks:

(i) the set of rules of biofilm developement must be formulated,

(ii) local grid refinement will lead to different system outputs, which is unphysical.

We are interested in simulating the biofilm growth problem as a *continuum without any probabilistic rules*. Since the interface is moving, the Finite Element Method may not provide accurate solutions for these kinds of problems. One of the advantages of meshfree methods is the ability to deal with evolving boundaries and moving interfaces, since in this case the discretization is being processed on a set of nodes only.

A number of various models dealing with biofilms has been proposed. The first mathematical model to simulate growth of bacterial colonies on the surface *as a continuum* is presented by Grimson and Barker [76] in 1993. Two years later Costerton publishes an overview of microbial biofilms [41], and in in 1998 Arshad et al. publish an overview of advances in fundamentals for biofilms modelling up to that time [10]. In the same year Picioreanu et al. use a combined approach, incorporating both a cellular automata and differential methods [168].

A numerical study in three dimensions of various processes in biofilms, such as spatial structure, conditions, and mass transfer, is conducted by Eberl et al. [51] in 2000. An important paper by Dockery and Klapper appears in 2001, wherein they explain the processes and phenomena of finger formation in biofilm structure [45]. In the same year Eberl et al. propose a deterministic continuum model for biofilm development [52], where a predator-pray model to simulate the biofilm growth using the singular diffusion coefficient as in [145] to distinguish between the biofilm and the surrounding fluid is utilized.

In 2002 Efendiev, Eberl et al. [53] prove existence of a solution and explain its longtime behaviour on an example of a nonlinear reaction-diffusion system in 2002 and van Loosdrecht et al. publish a paper on mathematical modelling of biofilm structures [195]. Chopp et al. propose a model of quorum sensing [38], which is often used in a number of later models.

Some similarities between process in biofilm growth and dynamics of tumour growth (see Section 7.4 on page 143) may be seen from the paper by Brú et al. [28]. A stochastic 3d simulation of biofilms [30] is suggested by Chang et al. in 2003. A rather simple Freter model of formation of biofilms is published by Jones et al. [99] in 2003. In the same year Chopp et al. investigate the dependence of quorum sensing and the depth of growing biofilm [39], and Eberl and Efendiev publish an overview of biofilm formation and decay modelling [50]. In 2004 Stoodley and Costerton publish a popular science essay [187] on effects of biofilms onto the natural environment and infectious diseases.

A three-dimensional continuum model of biofilm with individual cells [6] is proposed by Alpkvist et al. in 2006 and a model for heterogeneous biofilm development [5] in 2007. In 2006 Cogan proposes a two-fluid model description of biofilm disinfection [40]. In [47] the authors propose a mathematical model based on [38, 39], and use the X-FEM method to solve the physics of biofilm, and the level-set method [162, 181] to track the interface movement. For the solution of the level-set equation they use a finite difference grid.

Recently, in 2007, Ayati and Klapper suggest a multiscale model of biofilm as a senescence-structured fluid [15] with two spatial and time scales, where processes on a cell level are taken into account via a special variable to represent the cell senescence, i.e. the phenomenon when cells do not replicate themselves any longer. Towler et al. regard biofilm as a structure and propose a model of fluid-biofilm interaction via the Burger material law [192].

In the following we combine the developed strong form space-time meshfree method (STMCM) with the solution of a level-set equation in a monolithic way in both space and time, using the approach presented in Section 3 and focus on the validation of the presented space-time technique proposing a biofilm growth model.

7.3 A mathematical model of biofilm growth

The whole space-time domain of interest $Q = \Omega \times I$, where Ω is the spatial part, and $I = [0, T]$ the temporal one, consists of two parts: $Q = Q^{a,i} = Q^a \cup Q^i$, where Q^a is representing the biofilm itself – the *active* biofluid, and Q^i – representing the liquid substrate around – the *inactive* biofluid. In the proposed mathematical model the active biofluid is representing the biofilm and must model its behaviour, e.g. biofilm growth due to nutrients present in the domain, and the surrounding water, the inactive biofluid, is an auxiliary construction used for the solution strategy.

Figure 7.1 shows the two subdomains Ω_a and Ω_i of the active and inactive biofluids, respectively, at some moment in time $t = \text{const}$. The interface $\Sigma(t)$ at time t between Ω_a and Ω_i is described by a zero-level of a function ϕ, which is chosen to be negative in the domain Ω_a and positive in Ω_i. The interface between the active and inactive biofluids is assumed to be sharp which agrees with the experimental observations, since the length scales of the problems under consideration are much larger in comparison with the transition area. The detailed discussion of the interface description is given in Section 7.3.3.

7.3.1 Material description

In the proposed model the biofilm and the fluid around are considered to be incompressible fluids with different densities and viscosities. The unknowns to model the biofilm growth problem are: $\mathbf{v}^a, \mathbf{v}^i$ – the velocity fields of the active and inactive biofluid respectively, p^a, p^i – the respective pressures of the biofluids, the level set function ϕ which zero-level describes the interface between Q^a and Q^i, and the nutrient concentration n varying from zero to one.

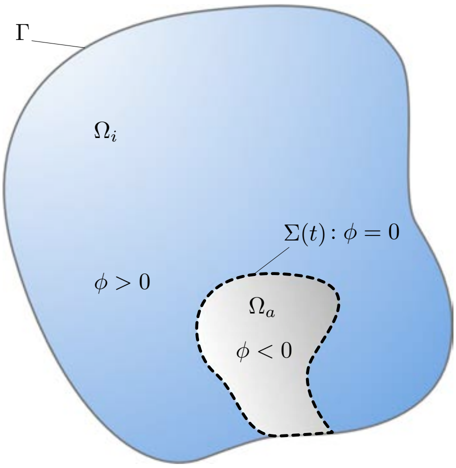


Figure 7.1 | Biofilm growth model setup.

In process of a biofilm growth the fluid velocities are very slow and inertial forces are negligible compared to viscous forces. Hence, in the proposed model the set of incompressible Stokes equations governs the behaviour of biofluids, whereas equation (7.1) is the balance of momentum of the active and inactive biofluids. The incompressibility condition is given by equation (7.2) and the initial and boundary conditions are equations (7.3)-(7.4). The source term f_g in the equation (7.2) is important for the *growth* of the active biofluid and it will be defined later, and the inactive biofluid is not growing but rather flows out through the outer boundary Γ. The Stokes equations can be seen as a special case of the Navier-Stokes equations describing incompressible Newtonian fluids, if inertial forces are negligible in comparison with viscous forces. This is a typical situation of slow viscous flows. The system of governing equations reads as

$$-\operatorname{div} \mathbf{T}^{a,i} = 0 \quad \text{in} \quad Q^{a,i}, \tag{7.1}$$

$$\rho^{a,i} \operatorname{div} \mathbf{v}^{a,i} = f_g \quad \text{in} \quad Q^{a,i}, \tag{7.2}$$

$$\mathbf{v}^{a,i}(\mathbf{x}, t = 0) = \mathbf{v}_0^{a,i} \quad \text{in} \quad Q^{a,i}_{t=0}, \tag{7.3}$$

$$\mathbf{v}^{a,i}(\mathbf{x}) = \bar{\mathbf{v}}^{a,i} \quad \text{on} \quad \Gamma, \tag{7.4}$$

with the Cauchy stress tensor $\mathbf{T}^{a,i} = -p^{a,i}\,\mathbf{I} + 2\mu^{a,i}\,\mathbf{D}^{a,i}$, describing the inner stress state of the active and inactive biofluids, and the fluid kinematics with the strain rate tensor $\mathbf{D}^{a,i} = \frac{1}{2}(\nabla\mathbf{v}^{a,i} + (\nabla\mathbf{v}^{a,i})^{\mathsf{T}})$.

The assumed material model of the biofilm is a Newtonian fluid with linear dependence between shear stresses and strain rate represented by the kinematic viscosity $\mu^{a,i}$. The stress state of a biofluid at rest is defined by the hydrostatic pressure $p^{a,i}$.

For an incompressible Newtonian fluid the equation (7.1) reads as

$$-\mu\nabla^2\mathbf{v}^{a,i} + \nabla p^{a,i} = 0. \tag{7.5}$$

- ▶ For fluids described by incompressible Navier-Stokes or stationary Stokes equations it is a well-known problem that spurious pressure oscillations occur in the numerical results unless some special care is taken. The reason is the violation of Ladyzhenskaya-Babuška-Brezzi (LBB) or 'inf-sup'-condition due to the lack of a separate equation for pressure. Different strategies are used to stabilize pressure including staggered grid approach, different orders for velocity and pressure approximations, see e.g. [61, 63, 64]. Another possibility is to add an artificial compressibility term to the incompressibility condition (7.2), which is used in the proposed model.

7.3.2 Nutrient description

Equation (7.6) is the advection-diffusion-reaction equation with a standard Monod reaction term f_{c} for nutrient *consumption* [52, 38, 39, 47] in the inactive biofluid only. In the active biofluid the nutrient of concentration n is consumed via the source term f_{g} in equation (7.2), which is responsible for the new biomass formation, i.e. the biofilm *growth*.

$$\begin{aligned} \frac{\partial n}{\partial t} + \mathbf{v}^{a,i}\cdot\nabla n - \nabla\cdot(d_n\nabla n) &= f_{\mathrm{c}} \quad \text{in } Q^{a,i}, && (7.6)\\ n(\mathbf{x}, t=0) &= n_0 \quad \text{in } Q^{a,i}_{t=0}, && (7.7) \end{aligned}$$

where d_n is the diffusion coefficient. The nutrients are being convected, i.e. transported through the domain, by the flow velocity field $\mathbf{v}^{a,i}$.

- ▶ As it was already mentioned, the nutrient concentration varies from zero to one. In order to avoid unphysical negative concentration values we use a cut-off function.

7.3.3 Interface description

Various techniques exist on how to track the moving interfaces [181, 161]. But perhaps one of the most successful ideas is the representation of the moving interface with help of the zero level of a high-dimensional level set function [181]. In the following we would like to briefly recall the level set procedure.

The level-set equation [181, 162] describes an evolution of a high-dimensional function ϕ, and the zero-level of this function ϕ is used for a description of a moving and deforming interface. But one does not have to know the position of the interface exactly. The function ϕ in the level-set equation is in most of the cases defined as a signed distance function from an arbitrary domain point to the interface. Thus, the different domains can be easily distinguished by simply looking at the sign of ϕ. In the cases where the velocity is known at the interface only the velocity extension technique is used [4]. In our model this is not necessary, since the velocity field is defined in the whole domain. A number of improved methods are available for interface capturing , e.g. a hybrid particle level-set method [55, 54] of Lagrangian type and [91], where a hybrid particle level-set method is combined with MLS approximants.

The interface between the active Q^a and the inactive Q^i biofluids is governed by the level-set equation (7.8) with the initial condition (7.9), which has been successfully used [161, 162, 181] for interface tracking in different problems with evolving interfaces, and allows to describe the interface implicitly.

$$
\begin{aligned}
\frac{\partial \phi}{\partial t} + \mathbf{v}^{a,i} \cdot \nabla \phi &= 0 \qquad \text{in} \quad Q^{a,i}, && (7.8)\\
\phi_0 &= \phi(\mathbf{x}, t=0) \qquad \text{in} \quad \Omega_{a,i}. && (7.9)
\end{aligned}
$$

The level-set function ϕ is chosen to be negative in the active biofluid domain Ω_a and positive in the inactive biofluid Ω_i, so that using the Heaviside function

$$
\mathsf{H}(\phi) = \begin{cases} 1, & \phi \geq 0 \\ 0, & \phi < 0 \end{cases}, \tag{7.10}
$$

one can easily distinguish between the inactive and active biofluids in an implicit way and the material constants (densities and viscosities) are defined as follows

$$
\begin{aligned}
\rho^{a,i} &= (1 - \mathsf{H}(\phi)) \cdot \rho^a + \mathsf{H}(\phi) \cdot \rho^i, && (7.11)\\
\mu^{a,i} &= (1 - \mathsf{H}(\phi)) \cdot \mu^a + \mathsf{H}(\phi) \cdot \mu^i, && (7.12)
\end{aligned}
$$

where ρ^a, μ^a are respectively the density and kinematic viscosity of the active biofluid, and ρ^i, μ^i – those of the inactive biofluid.

7.3.4 Nutrient consumption and biomass formation

The standard Monod term for nutrient consumption [52], which is made dependent on the level-set function ϕ in order to ensure that the nutrient is consumed in the active biofluid only, reads as

$$
f_c(\phi) = -(1 - \mathsf{H}(\phi)) \cdot \gamma_n \frac{n}{\eta_n + n}, \tag{7.13}
$$

with γ_n – specific growth rate, η_n – Monod reaction coefficient, and the biofilm growth term [52], i.e. the term responsible for the formation of a new biomass formation, is defined as

$$f_{\mathrm{g}}(\phi) = (1 - \mathsf{H}(\phi)) \cdot \gamma_n \frac{n}{\eta_n + n}. \tag{7.14}$$

7.3.5 Validation

In the proposed mathematical description for modelling a biofilm growth problem the sought unknowns are the level-set function ϕ, the nutrient concentration n, the velocities $\mathbf{v} = (v_1, v_2)^{\mathsf{T}}$ in x_1- and x_2-direction, respectively, and the pressure p all defined in a space-time slab $Q^{a,i}$.

If N is the number of points to discretize a space-time slab, then the system matrix for each space-time slab is of size $(5N \times 5N)$ and is structured as follows with bullets representing nonzero elements that contain appropriate derivatives of the IMLS kernel functions, the circle stands for a small artificial compressibility term, and stars denote the terms containing solutions from the previous time slab:

$$\begin{array}{|c|c|c|c|c|}\hline \bullet & & \bullet & \bullet & \\ \hline & \bullet & \bullet & \bullet & \\ \hline & & \bullet & \bullet & \bullet \\ \hline & & \bullet & \bullet & \bullet \\ \hline & & \bullet & \bullet & \circ \\ \hline \end{array} \cdot \begin{array}{|c|}\hline \phi \\ \hline n \\ \hline u \\ \hline v \\ \hline p \\ \hline \end{array} = \begin{array}{|c|}\hline \\ \hline \star \\ \hline \\ \hline \\ \hline \star \\ \hline \end{array} \tag{7.15}$$

The system of algebraic equations (7.15) is linearized and after application of initial and boundary conditions onto the system matrix (see [148] for a detailed description) the solution for the whole space-time slab is obtained, which then serves as the initial condition for the next space-time slab (see Section 3.6 on page 46 for the workflow).

- ▶ The system of discretized governing equations (7.15) is solved in a monolithic way, i.e. in one system, assuring all the physics is represented correctly. Usage of partitioned schemes for coupled problems is also possible, if needed, with minor changes to the code.

Parameters of the model used in the following validation examples are given in Table 7.1.

Unless stated differently, each of the time slabs in the following computations is discretized by $N = 4200$ points, and the IMLS-parameters are $\alpha = 8$, $q = 2$.

An isotropic biofilm growth

In the first example we investigate an isotropic biofilm growth behaviour in a stagnant fluid, i.e. without taking into account the outer velocity field in the inactive biofluid (see Figure

Table 7.1 | Parameters used for simulations of biofilm growth.

Parameter	Symbol	Units	Value	Source
Specific growth rate	γ_n	s^{-1}	0.417	[174]
Monod reaction coefficient	η_n	$mg{\cdot}l^{-1}$	0.100	[174]
Nutrient diffusion coefficient	d_n	$m^2{\cdot}s^{-1}$	9.67e-06	[174]
Ratio of viscosities	μ^a/μ^i	–	$1 \div 1000$	[40]
Ratio of densities	ρ^a/ρ^i	–	$1 \div 10$	assumed

7.2). The nutrient concentration supply is constant $n = n_0 = 1$ in the whole domain at $t = t_0 = 0$ and on the boundary Γ for all t. The nutrient is consumed by the active biofluid, its initial shape changes, and the inactive biofluid flows out through boundary Γ.

In Figure 7.3 on the facing page that shows the solution of this problem by STMCM, the dashed black line is the zero-level of the level-set function ϕ, i.e. the moving interface between the active and inactive biofluid, the nutrient concentration n is represented by different colours, and the vector plot shows the velocity field $\mathbf{v}$.

Initially, all velocities are zero and the nutrient concentration is constant in the whole computational domain. As the computation starts, nutrient diffuses into the active biofluid and is consumed by it, the nutrient consumption acts as a source term, and a velocity field is developing. Velocities act on the level-set function and the interface moves, as predicted, outwards. It must be noted, that the implicit tracking of the interface by the level-set function makes is easy to accomplish that the nutrient is consumed in the active biofluid only.

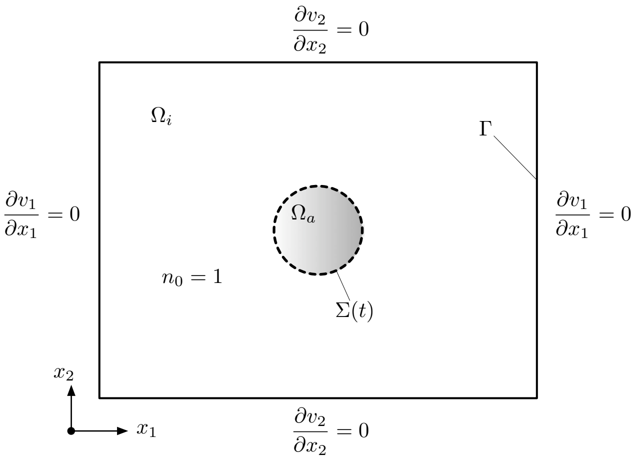


Figure 7.2 | An isotropic biofilm growth in a stagnant fluid.

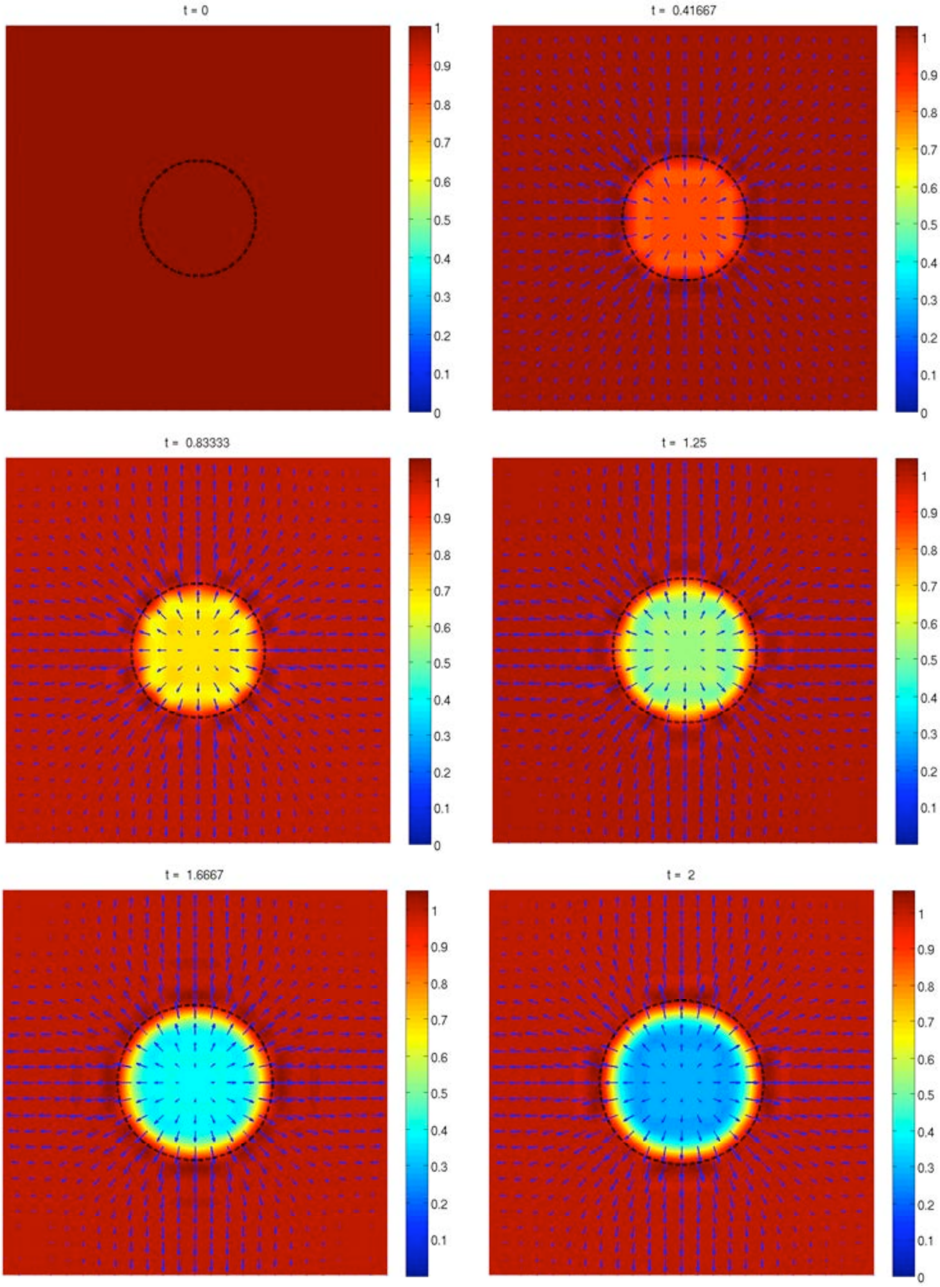


Figure 7.3 | Solution of the isotropic biofilm growth example by STMCM.

An anisotropic biofilm growth in a channel

The second example (see Figure 7.4 on the next page) models an anisotropic biofilm growth of several colonies in a channel with in- and outflow, on the channel walls non-slip conditions are implied. If the inactive biofluid is a stagnant one, the nutrient is not transported with the flow, but is diffused and consumed by the active biofluid, i.e. the biofilm, as in the previous example. Furthermore, we consider two configurations where individual colonies are located initially near to or far from to each other.

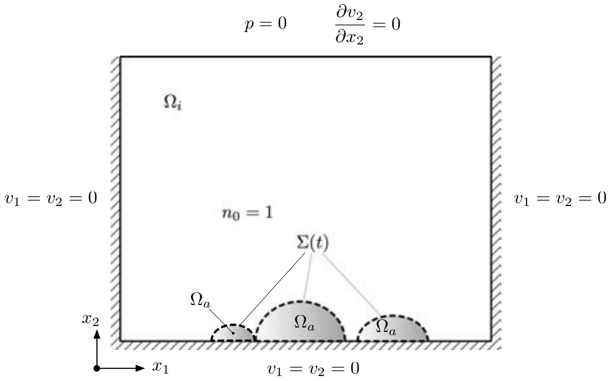
We note that in stagnant waters the biofilm tends to build mushroom- or finger-like structures. Whereas in a flow, the biofilm becomes streamlined to reduce drag. This observation is consistent with other reports, see e.g. [187, 45]. As long as nutrient is available in the whole system, the biofilm grows in all directions. However, with growing biomass the larger colonies start to dominate over the smaller ones due to their higher nutrient consumption and mushroom-like structures develop. The STMCM solution of the anisotropic biofilm growth example with several individual colonies located near to each other demonstrating the formation of mushroom-like structures is presented in Figure 7.5 on page 142.

The influence of the outer flow on the formation of the biofilm structure is demonstrated in Figures 7.6 and 7.7. In quiescent water, i.e. without an external velocity field, the biofilm colonies, initially placed far from each other, remain isolated and tend to form finger-like structures. If an external velocity field is present, the biofilm becomes a ‘streamliner’, i.e. it aligns with the streamlines of the flow to reduce drag. In high-velocity flows, the increased shear stresses may lead to detaching and sloughing of the biofilm [195].

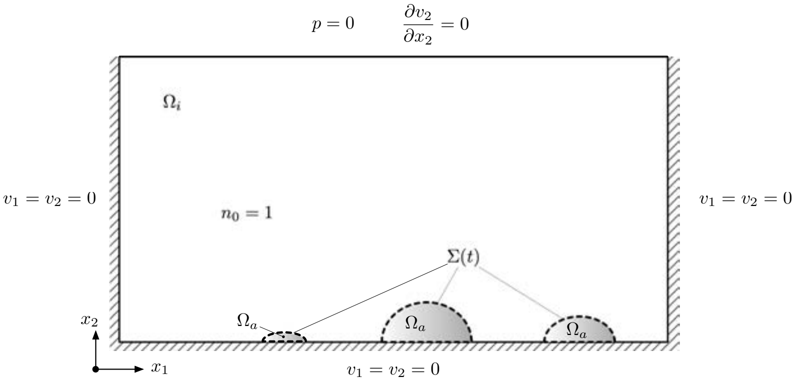
Figure 7.7 on page 144 demonstrates the influence of an external flow field with mean velocity in the middle of the channel $\overline{v} = 0.2$. Comparing it to the results obtained for a stagnant fluid shown in Figure 7.6 on page 143 the adaptation of the biofilm to the flow is observed.

Depending on parameters, the behaviour of biofilm growth processes might be different. Individual biofilm colonies might remain isolated and grow upwards, as it was shown by the STMCM solutions. Such a behaviour is observed for higher values of the specific growth rate γ_n and low values of the nutrient diffusion coefficient, see e.g. [45, 52]. For smaller values of γ_n, however, the biofilms also move horizontally so that the colonies merge, see e.g. [52].

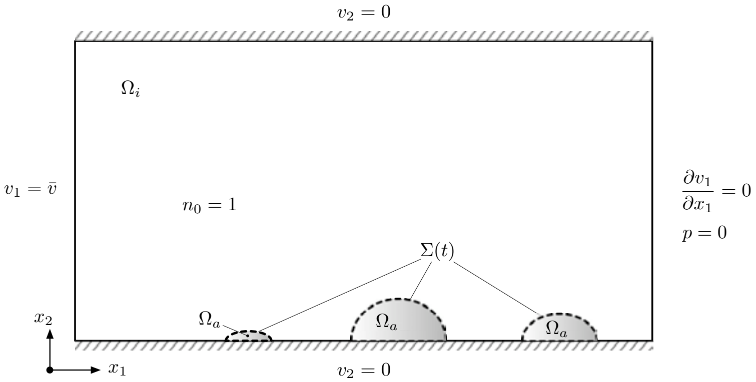
In order the presented model be used to simulate this kind of behaviour, certain rules for applying slip conditions at the channel walls must be incorporated, allowing tangential movement at the wall.



(a) *in a stagnant fluid (initial location near to each other)*



(b) *in a stagnant fluid (initial location far from each other)*



(c) *in a channel*

Figure 7.4 | An anisotropic biofilm growth of several colonies.

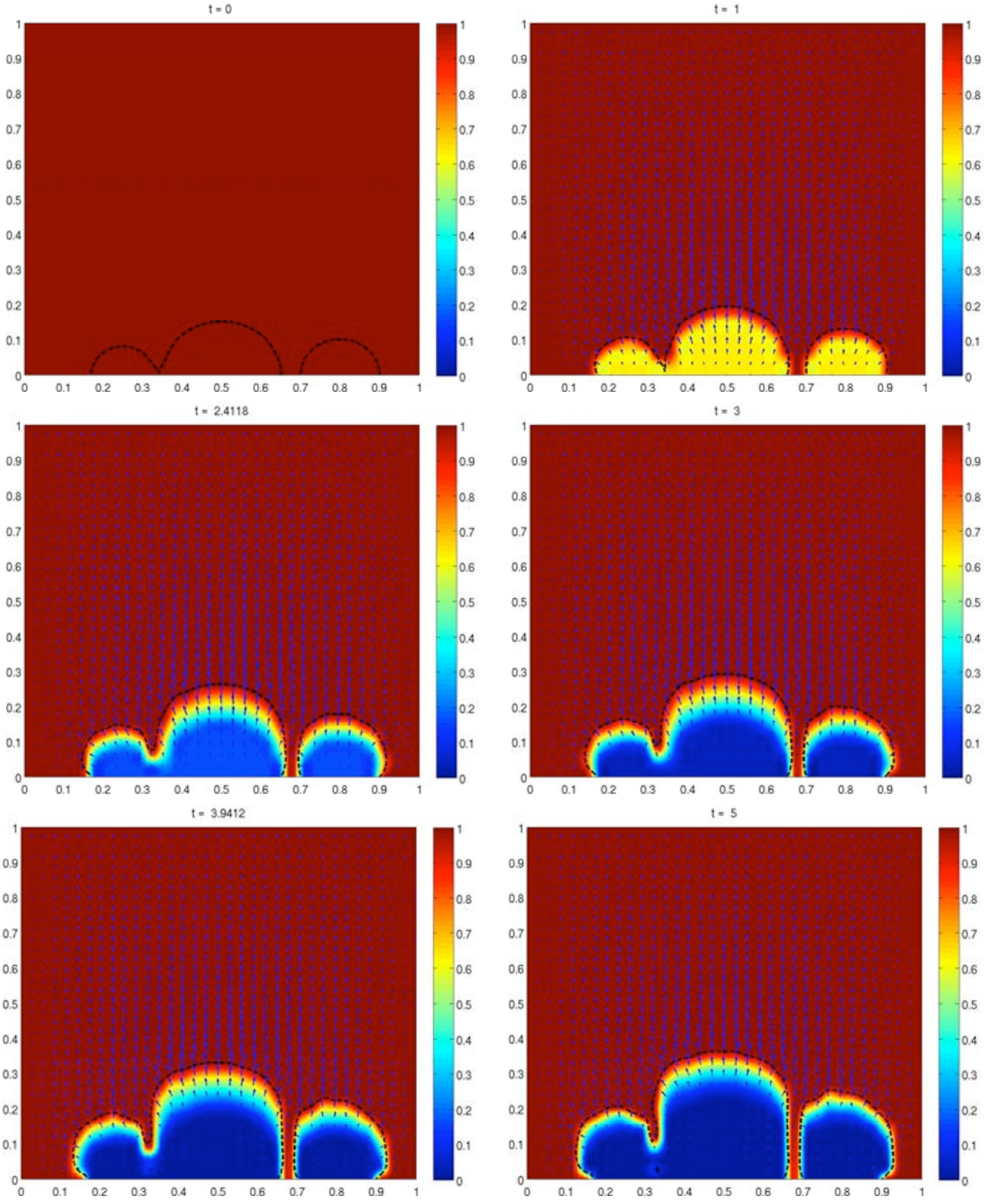


Figure 7.5 | STMCM solution of the anisotropic biofilm growth example in a stagnant fluid :: formation of mushroom-like structures.

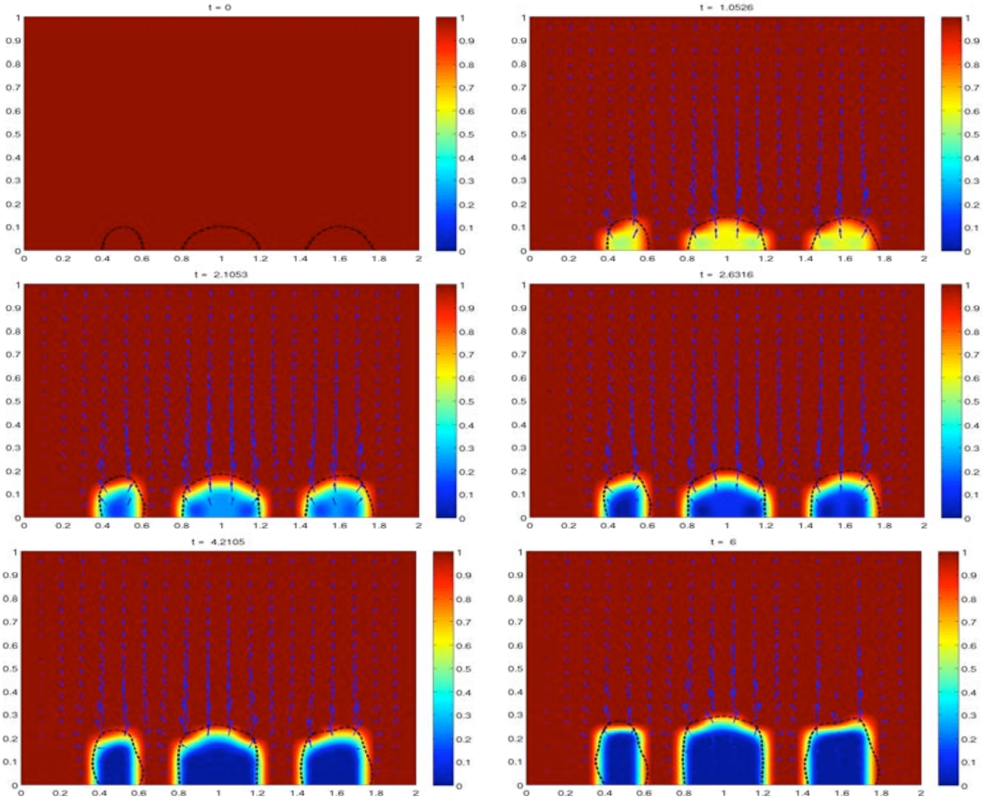


Figure 7.6 | STMCM solution of the anisotropic biofilm growth example in a stagnant fluid :: formation of finger-like structures.

7.4 Tumours

Tumour (from *lat.* tumere 'to swell') is called a swollen part of a body resulted due to an abnormal growth of tissue. In medicine one distinguishes between *benign* (harmless) and *malignant* (cancerous) tumours. Tumours can form in nearly all types of body cells, leading to a wide range of different tumour structures, and affect lives of practically every person. Nowadays there are hardly people who do not know someone (who maybe knows someone), who had cancer. Yet neither the causes of tumour formation, nor the effective and reliable medical treatments of this life-threatening disease are completely understood by or available to the modern science up to now.

We describe in this section the main features of genesis, development and growth of tumours following the representation given in [100]. In early stages of a tumour formation its growth is mainly affected by diffusion processes. For each cell the nutrients in the surrounding issue are diffused and wastes are transported from the tumour to the surrounding area. In this case one speaks of an *avascular* state and the tumour grows exponentially. However, this growth

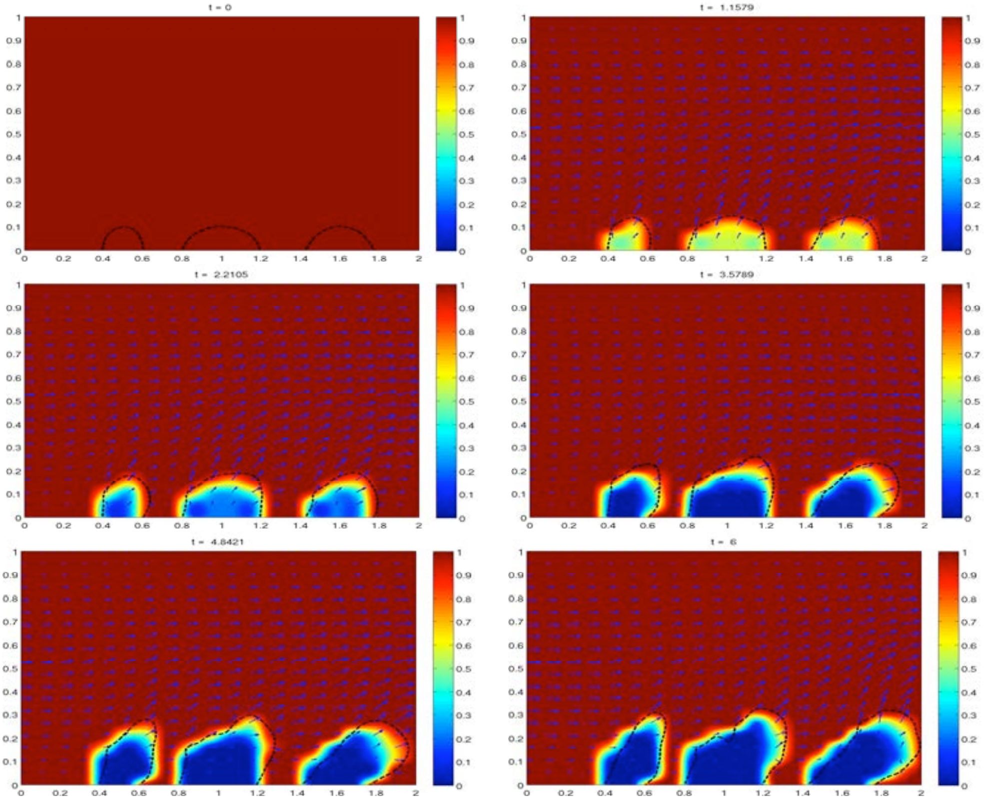


Figure 7.7 | STMCM solution of the anisotropic biofilm growth example in a channel flow :: influence of the flow on the formation of finger-like structures.

does not continue infinitely, since as nutrients are consumed, their concentration decreases from the boundary of the tumour to the centre until it falls below some critical value. The avascular state is, therefore, relatively harmless.

Due to angiogenesis, i.e. the development of new blood vessels and penetration of capillaries from the host tissue, the tumour may change into a *vascular* state developing the ability to send metastases to remote parts of body. Tumour cells produce a number of *tumour angiogenesis factors (TAF)* that quicken the formation of new capillaries.

Hence, after the tumour has reached the size limited by diffusion processes, the TAF start to act and perfusion, i.e. blood supply by circulating through vessels or other natural channels, replaces diffusion for the transport of nutrients and removal of wastes. In this case the tumour develops in an explosive manner. Breaking connections to the blood vessels would stop (or kill) the tumour growth.

7.4.1 A mathematical model of tumour growth

As in the case of biofilms, the tumour growth is the problem with moving boundaries which is difficult to solve with conventional mesh-based computational methods. The mathematical model we use here focuses on tracking the movement of the tumour boundary and examines the stability of the tumour. The stability is particularly important, as it leads either to a harmless non-malignant or a dangerous cancerous tumour.

In the following we present a model proposed in [75, 100] which deals with an avascular tumour growth process and, for some specific prescribed configurations, reduces to a non-linear ODE and can be used to validate the method presented in this thesis.

The model is rather flexible and can be adjusted to study various important situations, e.g. the tumour movements in case of presence of solid boundaries (e.g. an impermeable arterial wall), how the nutrients supply affects the growth of a tumour, the effect of presence of other tumour colonies, etc. A number of more sophisticated models of tumour growth are available, see e.g. [183, 182, 28].

The tumour grows is modelled as an incompressible fluid, i.e. the birth and death of cells produces internal pressure p which causes the motion of cells with velocity $\mathbf{v}$:

$$\mathbf{v} = -\nabla p. \tag{7.16}$$

The pressure p and nutrient concentration n satisfy diffusion equilibrium equations

$$\Delta p = S \quad \text{in } \Omega_a \tag{7.17}$$

and

$$\Delta n = 0 \quad \text{in } \Omega, \tag{7.18}$$

where S is a cell loss rate, Ω is the whole computational domain, and Ω_a is the active part of the domain, i.e. the tumour itself.

The following assumptions are made:

- The surface tension, and hence the pressure, is chosen proportional to the mean curvature $\varkappa$ on the boundary $\Gamma = \partial\Omega_a$ of the tumour

  $$p = \alpha\varkappa \quad \text{on } \Gamma, \tag{7.19}$$

  with some constant α.

- The pressure p and the tangential velocity component are chosen to be continuous on Γ, and the surrounding medium is considered sufficiently large, so that a constant nutrients supply is guaranteed.

- There is no diffusion transport from the interior if the nutrient concentration n falls below a critical level n_1.

- In the so called quiescent (i.e. inactive or dormant) region $n_1 > n > n_2$ and the thickness of this region is estimated from experiments [100]

$$h = \begin{cases} \nu\sqrt{n - n_1}, & n > n_1 \\ 0, & n < n_1, \end{cases}$$

where ν is a positive constant.

7.4.2 Validation :: a spherical avascular tumour

In order to use the presented model for validation, we use a simplified model of an isotropic growth of a spherical avascular tumour. In the case the tumour remains in its spherical shape with its outer boundary of a radius $R(t)$ while growing and the system of equations presented in the previous section reduces to the ODE (7.20). In this case one distinguishes between the proliferating cell region ($R_N < r < R$) and the necrotic core ($0 < r < R_N$).

The parameters of the model according to [100] are shown in Table 7.2 and they have the following physical meanings:

Table 7.2 | Parameters for the mathematical model of tumour growth.

S_1	S_2	λ	μ	n_1	n_2	n_∞
60	100	1	10	0.7	0.5	1

- S_1 is a cell loss rate due to *apoptosis*, i.e. programmed cell death, which is restricted to proliferated and quiescent regions,
- S_2 is a cell loss rate due to *necrosis*, i.e. traumatic cell death resulting from acute cell injury,
- n_1 and n_2 are the critical levels of nutrient concentration supply in proliferating and quiescent regions, respectively,
- n_∞ is the nutrient concentration in the faraway region, assuming the surrounding medium is much larger than the tumour size,
- λ and μ are constants, dependent on the ratio of the thickness of the outer shell and the size of the proliferation region.

The spherical growth of an avascular tumour is governed by the following nonlinear ODE (see [100] for a detailed derivation)

$$\frac{\mathrm{d}R}{\mathrm{d}t} - (\alpha_1 + \alpha_3\alpha_4)R = \alpha_2 \frac{\left(\alpha_4 R^3 + \alpha_5 R^2 \sqrt{\alpha_6 R^2 + \alpha_7}\right)^3}{R} + \alpha_8\sqrt{\alpha_6 R^2 + \alpha_7}, \tag{7.20}$$

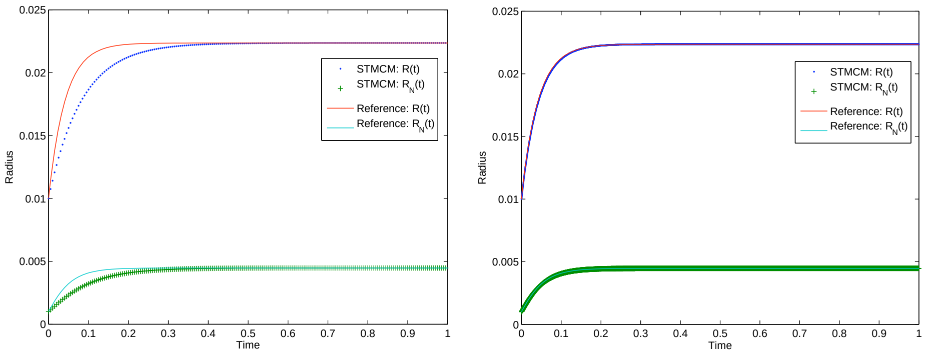
where coefficients $\alpha_1,\dots,\alpha_7$ are computed from the parameters of the model as:

$$\begin{aligned} \alpha_1 &= -\frac{S_1}{3}, & \alpha_5 &= \frac{\mu}{2(n_\infty - n_2)}, \\ \alpha_2 &= \frac{S_1 - S_2}{3}, & \alpha_6 &= \mu^2, \\ \alpha_3 &= -\frac{\lambda}{\mu}(n_2 - n_\infty), & \alpha_7 &= 4(n_\infty - n_1), \\ \alpha_4 &= -\frac{\mu^2}{2(n_\infty - n_2)}, & \alpha_8 &= \alpha_3\alpha_5. \end{aligned} \tag{7.21}$$

The radius of the necrotic core can be computed from the radius of the outer boundary of the tumour, i.e. from the solution $R(t)$ of the equation (7.20), as

$$R_N = \alpha_4 R^3 + \alpha_5 R^2 \sqrt{\alpha_6 R^2 + \alpha_7}. \tag{7.22}$$

Figure 7.8 shows the solution to the tumour growth problem (7.20) and (7.22) by STMCM with different parameters, resolving the nonlinearities by a Picard fixed point iteration scheme for each of the time slabs (see Section 3.3). Comparing our solution with a reference solution obtained by a multivalue variable-order Adams-Bashforth-Moulton predictor-corrector (ABM-PECE) method we obtain a good agreement, see Figure 7.8(b).



(a) $N = 2$ (b) $N = 33$

Figure 7.8 | Solutions of the tumour growth problem computed by STMCM with the time-slab thickness $\delta_n = 0.01$. Each of the time slabs is discretized by N Halton points.

Due to high nonlinearities in the system each of the time slabs of thickness $\delta_n = 0.01$ was discretized by $N = 33$ Halton points (compare with Figure 7.8(a), where each of the time slabs was discretized by $N = 3$ Halton points).

The expected behaviour of an avascular state has been observed: at the beginning of the growth process the diffusion processes prevails and the tumour is growing exponentially. Eventually, the limited size of the tumour is achieved and the tumour is not growing any longer. For the coarse discretization of each time slab with only $N = 3$ Halton points the STMCM does not catch entirely the formation process, yet it does reach the limited tumour size correctly. For the finer discretization with $N = 33$ Halton points the results are in excellent agreement with the reference solution.

Convergence behaviour of the STMCM solution compared to the reference one is demonstrated in Figure 7.9 and Table 7.3 for $\alpha = 8$, $\rho = 3h_{\Omega,\mathscr{S}}$, $q = 1$ and $\delta_n = 0.01$. We obtain the expected second order convergence for averaged absolute and relative errors.

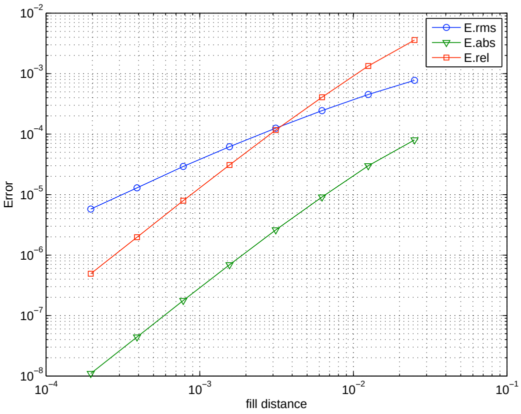


Figure 7.9 | Convergence study: tumour growth model.

Table 7.3 | Convergence results the tumour growth model.

Δt	E_{rms}	E_{abs}	E_{rel}	EOC_{rms}	EOC_{abs}	EOC_{rel}
2.5000e-02	7.7548e-04	8.0134e-05	3.5841e-03	–	–	–
1.2500e-02	4.5205e-04	2.9922e-05	1.3383e-03	0.7786	1.4212	1.4212
6.2500e-03	2.4432e-04	9.1026e-06	4.0712e-04	0.8877	1.7168	1.7168
3.1250e-03	1.2540e-04	2.6080e-06	1.1665e-04	0.9622	1.8033	1.8033
1.5625e-03	6.1880e-05	6.8979e-07	3.0851e-05	1.0190	1.9187	1.9187
7.8125e-04	2.9201e-05	1.7742e-07	7.9350e-06	1.0835	1.9590	1.9590
3.9063e-04	1.2946e-05	4.4137e-08	1.9741e-06	1.1735	2.0071	2.0071
1.9531e-04	5.7684e-06	1.1007e-08	4.9231e-07	1.1662	2.0035	2.0035

“Whenever man comes up with a better mousetrap, nature immediately comes up with a better mouse.”

-- James Carswell

“Here is a test to find whether your mission on Earth is finished: If you’re alive, it isn’t. ”

-- Richard Bach

CHAPTER 8

Summary and Outlook

8.1 Summary

Numerical solution of coupled engineering problems with evolving boundaries on computational domains with irregular shapes is still a challenging task in engineering practice. In this thesis an innovative numerical approach for solving ordinary and partial differential equations by a consistent discretization in space and time was presented.

The developed Space-Time Meshfree Collocation Method (STMCM), based upon the Interpolating Moving Least Squares (IMLS) technique, uses a strong formulation of governing equations and belongs to the class of meshfree methods which results in a number of advantages compared to established mesh-based methods, especially for coupled problems with moving interfaces. In the proposed approach the numerical discretization is performed on a set of nodes only, i.e. without an a priori node-to-node connectivity in form of a mesh. Furthermore, due to the usage of the strong form of equations there is no need to evaluate integrals. The method is, therefore, truly meshfree since no mesh at all is required: neither for field variables, nor for a numerical integration. Thus, the STMCM can be implemented easily and straightforwardly and computations are less expensive than with other meshfree methods.

A rigorous mathematical derivation of avoiding the singularity-by-construction problem that arises in IMLS has been given. Using the IMLS method a ‘mapping’ technique for a non-trivial distribution of computational points within irregularly-shaped domains, which is a part of a workflow of any meshfree method, has been constructed based on an implicit representation of surfaces that interpolate scattered point cloud data in multiple dimensions. Moreover, formulae for higher-order derivatives of the IMLS kernel functions needed for the strong-form discretization of differential equations have been obtained. Due to the interpolating character of the IMLS kernel functions, the Kronecker delta property is fulfilled and the implementation of essential boundary conditions is simplified.

Numerical studies of robustness, accuracy, and convergence of the method have been performed applying it to various types of differential equations with known either analytical or reference solutions. It has been shown that with the STMCM a simple yet powerful tool of numerical analysis is available for obtaining accurate solutions of linear and nonlinear ordinary and partial differential equations, which has been demonstrated on a number of examples including, amongst others, Lorenz equations, diffusions, vibrations, convection processes, boundary layers, etc. To demonstrate the applicability of the STMCM to problems with evolving boundaries and changing topologies two applications from the fields of environmental and medical engineering, respectively, have been studied. Both problems involve moving and deforming interfaces and are, therefore, difficult to solve by established mesh-based methods.

In the first application a mathematical model based on a two-biofluid representation of a biofilm as a continuum has been proposed whereas the incompressible Stokes equations with supplementary terms, which are responsible for the formation of new biomass, govern the flow of the biofluids. The moving interface is being implicitly described by the level-set equation, and an advection-diffusion-reaction equation is used for nutrient description. Several examples of biofilm growth validating the model have been considered. It was shown that the proposed model is capable of reproducing such phenomena like finger- and mushroom-like structures observed in experiments and the results are consistent with those obtained by other biofilm models. In the second application the STMCM has been applied to an existing mathematical model of tumour growth and the ability of the method to give accurate and convergent results has been demonstrated on an example of the growth of an avascular tumour.

8.2 Outlook

In the presented implicit surface reconstruction technique used to design a mapping for the distribution of computational points within domains with irregular boundaries, the higher-dimensional level-set function ϕ is defined within the bounding box of the scattered data. Especially in three and more dimensions (e.g. three spatial dimensions and the temporal one), it can be expensive to evaluate the level-set function on a set of points covering the whole bounding box. The computational cost can be reduced significantly if specific techniques are applied. Since only a zero-isolevel of the level-set function is of interest, it is sufficient to construct ϕ only in the vicinity of its zero-isolevel. A promising approach is the adaptive meshfree technique by Behrens et al. [18] as well as the meshfree thinning techniques in 3d by Dyn et al. [49], based on multiresolution methods for modelling of scattered data by Iske [96], and the hybrid particle level set method for improved capturing of interfaces by Enright et al. [54]. The proposed model for biofilm growth can be used further for a deeper study of phenomena and mechanisms occurring in biofilms. An important part would be a search for suitable parameters enabling the model to describe biofilm decay, detachment and sloughing, as well as incorporating certain slip conditions for biofilms at surfaces they are attached to. A good starting point for this purpose would be the papers by Eberl et al. [50], van Loosdrecht et al. [195], and the recent works by Alpkvist et al. [6, 5].

APPENDIX A

Analytical expressions

In this chapter we provide the formulae used to compute the exact values of functions and their derivatives, which are used to test the proposed approach for interpolation of scattered data in Section 5 on page 63, as well as the surfaces and their normals used in the interpolation of 3d point cloud data in Section 4.3.2 on page 55.

Franke's function

Franke's function is a standard test function for scattered data interpolation in 2d and its formula is given by

$$\begin{aligned} F(x,y) = &-\frac{1}{5}\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2}+\frac{1}{2}\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)} \\ &+\frac{3}{4}\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}+\frac{3}{4}\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}\,. \end{aligned} \tag{A.1}$$

The formulae for the first and second derivatives of the Franke's function w.r.t. to space coordinates x and y are obtained as follows:

$$
\begin{aligned}
\frac{\partial F(x,y)}{\partial x} =& -2.25\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}(-7+9x)\\
&+3.6\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2}(-4+9x)\\
&-3.375\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}(-2+9x)\\
&-0.27551\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}(1+9x),
\end{aligned}
\tag{A.2}
$$

$$
\begin{aligned}
\frac{\partial F(x,y)}{\partial y} =& 3.6\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2}(-7+9y)\\
&-2.25\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}(-3+9y)\\
&-3.375\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}(-2+9y)\\
&-1.35\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}(1+9y),
\end{aligned}
\tag{A.3}
$$

$$
\begin{aligned}
\frac{\partial^2 F(x,y)}{\partial x^2} =& 32.4\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2} - 20.25\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}\\
&-30.375\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}\\
&-2.47959\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}\\
&+10.125\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}(-7+9x)^2\\
&-64.8\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2}(-4+9x)^2\\
&+15.1875\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}(-2+9x)^2\\
&+0.101208\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}(1+9x)^2,
\end{aligned}
\tag{A.4}
$$

$$
\begin{aligned}
\frac{\partial^2 F(x,y)}{\partial y^2} =& 32.4\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2} - 20.25\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}\\
&-30.375\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}\\
&-12.15\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2} - 64.8\,\mathrm{e}^{-(-4+9x)^2-(-7+9y)^2}(-7+9y)^2\\
&+10.125\,\mathrm{e}^{\frac{1}{4}\left(-(-7+9x)^2-(-3+9y)^2\right)}(-3+9y)^2\\
&+15.1875\,\mathrm{e}^{\frac{1}{4}\left(-(-2+9x)^2-(-2+9y)^2\right)}(-2+9y)^2\\
&+2.43\,\mathrm{e}^{-\frac{1}{49}(1+9x)^2-\frac{1}{10}(1+9y)^2}(1+9y)^2.
\end{aligned}
\tag{A.5}
$$

The graphical representation of Franke's function and its derivatives is provided in Figure 5.1 on page 66.

“Blupp” test function

$$
\begin{aligned}
z(x,y) &= \cos(x^2+y^2) && \text{(A.6)}\\
\frac{\partial z(x,y)}{\partial x} &= -2x\sin\left(x^2+y^2\right), && \text{(A.7)}\\
\frac{\partial z(x,y)}{\partial y} &= -2y\sin\left(x^2+y^2\right), && \text{(A.8)}\\
\frac{\partial^2 z(x,y)}{\partial x^2} &= -4x^2\cos\left(x^2+y^2\right)-2\sin\left(x^2+y^2\right), && \text{(A.9)}\\
\frac{\partial^2 z(x,y)}{\partial y^2} &= -4y^2\cos\left(x^2+y^2\right)-2\sin\left(x^2+y^2\right). && \text{(A.10)}
\end{aligned}
$$

The graphical representation of this function and its derivatives is provided in Figure 5.4 on page 70.

Helix

A helix surface is given in parametric representation by

$$\mathbf{r}=\{(R+r\cos u)\cos v;\ (R+r\cos u)\sin v;\ r\sin u+hv\}, \tag{A.11}$$

and its normal is computed as $\mathbf{n}=(n_x,n_y,n_z)^{\mathsf{T}}=\left|\frac{\partial\mathbf{r}}{du}\times\frac{\partial\mathbf{r}}{dv}\right|$, so that

$$
\begin{aligned}
n_x &= -rh\sin u\sin v-(R+r\cos u)r\cos v\cos u, && \text{(A.12)}\\
n_y &= rh\sin u\cos v-(R+r\cos u)r\cos u\sin v, && \text{(A.13)}\\
n_z &= (R+r\cos u)\cdot(-r\sin u\cos^2 v-r\sin u\sin^2 v). && \text{(A.14)}
\end{aligned}
$$

The helix surface for $u=0,\ldots,2\pi$, $v=0,\ldots,4\pi$, $R=0.6$, $r=0.2$, and $h=0.25$ is shown in Figure 4.13 on page 60.

Bessel functions of the first kind

In Chapter 6 the so called Bessel functions of the first kind of a certain order are used as initial conditions for heat and wave equations. This specific choice makes it possible to obtain analytical solutions and to study the accuracy and convergence behaviour of the method presented in this thesis.

Bessel functions are the solution of a linear second order Bessel ODE

$$x^2 \frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x^2 - n^2)y = 0, \qquad \text{(A.15)}$$

where $n > 0$ is a constant; often (but not necessarily) $n \in \mathbb{N}$.

The Bessel functions of the first kind of order n are defined [3] as

$$\mathrm{J}_n(x) := \sum_i^{\infty} (-1)^i \frac{(x/2)^{n+2i}}{i!(n+i)!}. \qquad \text{(A.16)}$$

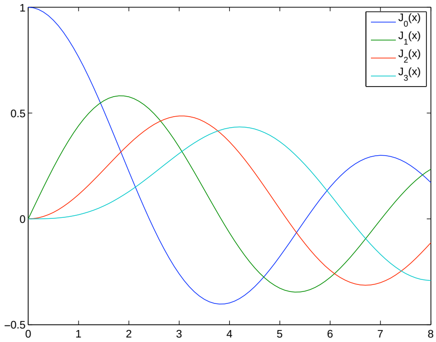


Figure A.1 | Bessel functions of the first kind.

Figure A.1 shows the Bessel functions of the first kind exemplarily for $n = 0, \ldots, 4$. For $n = 0$ the first two roots of the Bessel function of the first kind of order zero, i.e. $\mathrm{J}_0(x)$ are $\zeta_1 = 2.4048$, $\zeta_2 = 5.5201$, see [3] for details.

APPENDIX B

Invariance of the Laplacian

Laplacian, or Laplace operator, Δ is invariant w.r.t. translations and rotations and in this chapter we derive formulae for it in polar coordinates used in Section 6.6.

The Laplacian is defined in Cartesian coordinates x and y as

$$\Delta_{(x,y)} := \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \tag{B.1}$$

The relationship between the Cartesian (x, y) and polar (r, θ) coordinates is given as

$$x = r\cos\theta \quad \text{and} \quad y = r\sin\theta,$$

and the corresponding Jacobian

$$\mathrm{J} = \begin{pmatrix} \frac{\partial x}{\partial r} & \frac{\partial y}{\partial r} \\ \frac{\partial x}{\partial \theta} & \frac{\partial y}{\partial \theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -r\sin\theta & r\cos\theta \end{pmatrix} \tag{B.2}$$

and its inverse

$$\mathrm{J}^{-1} = \begin{pmatrix} \frac{\partial r}{\partial x} & \frac{\partial \theta}{\partial x} \\ \frac{\partial r}{\partial y} & \frac{\partial \theta}{\partial y} \end{pmatrix} = \begin{pmatrix} \cos\theta & -\frac{1}{r}\sin\theta \\ \sin\theta & \frac{1}{r}\cos\theta \end{pmatrix}. \tag{B.3}$$

Applying the chain differentiation rule we obtain

$$\frac{\partial^2}{\partial x^2} = \frac{\partial}{\partial x} \cdot \frac{\partial}{\partial x} = \left(\cos\theta \frac{\partial}{\partial r} - \frac{1}{r}\sin\theta \frac{\partial}{\partial \theta}\right)^2 \tag{B.4}$$

and

$$\frac{\partial^2}{\partial y^2} = \frac{\partial}{\partial y} \cdot \frac{\partial}{\partial y} = \left(\sin\theta \frac{\partial}{\partial r} + \frac{1}{r}\cos\theta \frac{\partial}{\partial \theta}\right)^2. \tag{B.5}$$

On opening the brackets and summing up (B.5) and (B.4) we immediately get the Laplacian in polar coordinates

$$\Delta_{(r,\theta)} = \Delta_{(x,y)} =: \Delta = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \theta^2}. \tag{B.6}$$

In a radially symmetric case, i.e. when the function does not depend on the angle θ (and hence the derivatives w.r.t. θ are zero), the Laplacian reduces to

$$\Delta_{(r)} = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}. \tag{B.7}$$

Bibliography

[1] The biofilm hypertextbook `www.erc.montana.edu/biofilmbook`.

[2] The Stanford 3D scanning repository `graphics.stanford.edu/data/3dscanrep`.

[3] Abramowitz, M., and Stegun, I. A. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, 9th Dover printing, 10th GPO printing ed. Dover, 1964.

[4] Adalsteinsson, D., and Sethian, J. A. The fast construction of extension velocities in level set method. *Journal of Computational Physics 148* (1999), 2–22.

[5] Alpkvist, E., and Klapper, I. A multidimensional multispecies continuum model for heterogeneous biofilm development. *Bulletin of Mathematical Biology 69* (2007), 765–789.

[6] Alpkvist, E., Picioreanu, C., van Loosdrecht, M., and Heyden, A. Three-dimensional biofilm model with individual cells and continuum eps matrix. *Biotechnology and Bioengineering 94*, 5 (2006), 961–979.

[7] Aluru, N. R. A point collocation method based on reproducing kernel approximations. *International Journal for Numerical Methods in Engineering 47* (2000), 1083–1121.

[8] Ames, W. F. *Numerical methods for partial differential equations*, second ed. Academic Press, New York, 1977. Computer Science and Applied Mathematics, Applications of Mathematics Series.

[9] Armentano, M., and Durán, R. Error estimates for moving least square approximations. *Applied Numerical Mathematics 37* (2001), 397–416.

[10] Arshad, M., Chaudrhry, S., and Beg, S. A review on the mathematical modeling of bioflim processes: advances in fundamentals of biofilm modeling. *Chemical Engineering Technology 21*, 9 (1998), 701–710.

[11] Atluri, S., Cho, J., and Kim, H. Analysis of thin beams, using the meshless local Petrov-Galerkin method, with generalized moving least squares interpolants. *Computational Mechanics 24* (1999), 334–347.

[12] Atluri, S., and Zhu, T. A new meshless Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics 22* (1998), 117–127.

[13] Atluri, S., and Zhu, T. The meshless local Petrov-Galerkin (MLPG) approach for solving problems in elasto-statics. *Computational Mechanics 25* (2000), 169–179.

[14] Atluri, S., and Zhu, T. New concepts in meshless methods. *International Journal for Numerical Methods in Engineering 47* (2000), 537–556.

[15] Ayati, B., and Klapper, I. A multiscale model of biofilm as a senescence-structured fluid. *Multiscale modelling and simulation 6*, 2 (2007), 347–365.

[16] Babuška, I., and Melenk, J. The partition of unity method. *International Journal for Numerical Methods in Engineering 40* (1997), 727–758.

[17] Baker, G. L., and Blackburn, J. A. *The Pendulum: a case study in physics*. Oxford University Press, 2005.

[18] Behrens, J., Iske, A., and Käser, M. Adaptive meshfree method of backward characteristics for nonlinear transport equation. In *Meshfree methods for partial differential equations*,

M. Griebel and M. A. Schweitzer, Eds., vol. 26 of *Lecture Notes in Computational Science and Engineering*. Springer, 2002.

[19] BELYTSCHKO, T., GUO, Y., LIU, W., AND XIAO, S. A unified stability analysis of meshless particle methods. *International Journal for Numerical Methods in Engineering 48* (2000), 1359–1400.

[20] BELYTSCHKO, T., KRONGAUZ, Y., FLEMING, M., ORGAN, D., AND KAM, L. W. Smoothing and accelerated computations in the element free Galerkin method. *J. Comput. Appl. Math. 74* (1996), 111–126.

[21] BELYTSCHKO, T., KRONGAUZ, Y., ORGAN, D., FLEMING, M., AND KRYSL, P. Meshless methods: An overview and recent developments. *Comput. Methods Appl. Mech. Engng 139* (1996), 3–47.

[22] BELYTSCHKO, T., LU, Y., AND GU, L. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering 37* (1994), 229–256.

[23] BENTLEY, J. Multidimensional binary search trees used for associative searching. *Comm. ACM 18* (1975), 509–517.

[24] BJÖRCK, B., AND PEREYRA, V. Solution of Vandermonde systems of equations. *Mathematics of computation 24*, 112 (1970), 893–903.

[25] BOYD, J. P. *Chebyshev & Fourier spectral methods*. Dover Publishers, Berlin, 1989. Computer Science and Applied Mathematics, Applications of Mathematics Series.

[26] BREITKOPF, P., RASSINEUX, A., TOUZOT, G., AND VILLON, P. Explicit form and efficient computation of MLS shape functions and their derivatives. *International Journal for Numerical Methods in Engineering 48*, 3 (2000), 451–466.

[27] BREUSS, M., DOLEJŠI, V., AND MEISTER, A. Anisotropic adaptive resolution of boundary layers for heat conduction problems. *ZAMM Z. Angew. Math. Mech. 86*, 6 (2006), 450–463.

[28] BRÚ, A., ALBERTOS, S., SUBIZA, J., GARCIA-ASENJO, J., AND BRÚ, I. The universal dynamics of tumor growth. *Biophysical Journal 85* (2003), 2948–2961.

[29] CARR, J., BEATSON, R., AND CHERIE, J. E. A. Reconstruction and representation of 3d objects with radial basis functions. *SIGGRAPH* (2001), 67–76.

[30] CHANG, I., GILBERT, E., ELIASHBERG, N., AND KEASLING, J. A three-dimensional, stochastic simulation of biofilm growth and transport-related factors that affect structure. *Journal of Industrial Microbiology 149* (2003), 2859–2871.

[31] CHEN, J.-S., HAN, W., YOU, Y., AND MENG, X. A reproducing kernel method with nodal interpolation property. *Internat. J. Numer. Methods Engrg. 56*, 7 (2003), 935–960.

[32] CHEN, J.-S., HU, W., AND HU, H. Reproducing kernel enhanced local radial basis collocation method. *International Journal for Numerical Methods in Engineering* (2007).

[33] CHEN, J.-S., AND WANG, H.-P. New boundary condition treatments in meshfree computation of contact problems. *Comput. Methods Appl. Mech. Engrg. 187*, 3-4 (2000), 441–468.

[34] CHEN, W. Boundary knot method: a meshless, exponential convergence, integration-free, and boundary-only RBF technique. *Computers and Mathematics with applications 43* (2002), 379–391.

[35] CHEN, W. Symmetric boundary knot method. *Engng. Anal. Bound. Elem. 26*, 6 (2002), 489–494.

[36] CHEN, W., AND HON, Y. Numerical convergence of boundary knot method in the analysis of Helmholtz, modified Helmholtz, and convection-diffusion problems. *Computer Methods in Applied Mechanics and Engineering 192* (2003), 1859–1875.

[37] CHENG, A., AND CABRAL, J. Direct solution of ill-posed boundary value problems by radial basis function collocation method. *International Journal for Numerical Methods in Engineering 64*, 1 (2005), 45–64.

[38] CHOPP, D., KIRISITS, M., MORAN, B., AND PARSEK, M. A mathematical model of quorum sensing in a growing biofilm. *Journal of Industrial Microbiology and Biotechnology 29*, 6 (2002), 339–346.

[39] CHOPP, D., KIRISITS, M., MORAN, B., AND PARSEK, M. The dependence of quorum sensing on the depth of a growing biofilm. *Bulletin of Mathematical Biology 65* (2003), 1053–1079.

[40] COGAN, N. Two-fluid model of biofilm disinfection. Tech. rep., Florida State University, Department of Mathematics, 2006.

[41] COSTERTON. Overview of microbial biofilms. *Journal of Industrial Microbiology 15* (1995), 137–140.

[42] COURANT, R., FRIEDRICHS, K., AND LEWY, H. Über die partiellen Differenzengleichungen der mathematischen Physik. *Math. Ann. 100*, 1 (1928), 32–74.

[43] DEMKOWICZ, L., KARAFIAT, A., AND LISZKA, T. On some convergence results for FDM with irregular mesh. *Computer Methods in Applied Mechanics and Engineering 42* (1984), 343–355.

[44] DESIMONE, H., URQUIZA, S., ARRIETA, H., AND PARDO, E. Solution of Stokes equations by moving least squares. *Comm. Numer. Methods Engrg. 14*, 10 (1998), 907–920.

[45] DOCKERY, J., AND KLAPPER, I. Finger formation in biofilm layers. *SIAM Journal of Applied Mathematics 62*, 3 (2001), 853–869.

[46] DUARTE, C., AND ODEN, J. An h-p adaptive method using clouds. *Computer Methods in Applied Mechanics and Engineering 139* (1996), 237–262.

[47] DUDDU, R., BORDAS, S., CHOPP, D., AND MORAN, B. A combined extended finite element and level set method for biofilm growth. *International Journal for Numerical Methods in Engineering 2* (2007), 1–33.

[48] DUFLOT, M. A meshless method with enriched weight functions for three-dimensional crack propagation. *International Journal for Numerical Methods in Engineering 65* (2006), 1970–2006.

[49] DYN, N., ISKE, A., AND WENDLAND, H. Meshfree thinning of 3d point clouds. *Foundations of Computational Mathematics 8* (2008), 409–425.

[50] EBERL, H., AND EFENDIEV, M. Modeling, analysis, and simulation of biofilm formation and decay. *PAMM 2*, 1 (2003), 430–431.

[51] EBERL, H., PICIOREANU, C., HEIJNEN, J., AND VAN LOOSDRECHT, M. A three-dimensional numerical study on the correlation of spatial structure, hydrodynamic conditions, and mass transfer and conversion in biofilms. *Chemical Engineering Science 55*, 24 (2000), 6209–6222.

[52] EBERL, H. J., PARKER, D. F., AND VAN LOOSDRECHT, M. C. A new deterministic spatio-temporal continuum model for biofilm development. *Journal of Theoretical Medicine 3* (2001), 161–175.

[53] EFENDIEV, M., EBERL, H., AND ZELIK, S. Existence and longtime behavior of solutions of a nonlinear reaction-diffusion system arising in the modeling of biofilms. In *RIMS Kyoto Kokyuroko* (2002), vol. 1258.

[54] ENRIGHT, D., FEDKIW, R., FERZIGER, J., AND MITCHELL, I. A hybrid particle level set method for improved interface capturing, 2002.

[55] ENRIGHT, D., LOSASSO, F., AND FEDKIW, R. A fast and accurate semi-Lagrangian particle level set method, 2004.

[56] FASSHAUER, G. Solving partial differential equations by collocation with radial basis functions. In *Proceedings of Chamonix 1996* (1997), Vanderbilt Univ. Press.

[57] FASSHAUER, G. Approximate moving least-squares approximation for time-dependent PDEs. In *Fifth World Congress on Computational Mechanics (WCCM V)* (2002).

[58] FASSHAUER, G. *Approximation Theory X: Wavelets, Splines, and Applications*. Vanderbilt Univ. Press, 2002, ch. Matrix-free multilevel moving-least squares methods, pp. 271–281.

[59] FASSHAUER, G. E. *Meshfree approximation methods with Matlab*. World Scientific Publishing Co., 2007.

[60] FERNÁNDEZ-MÉNDEZ, S., AND HUERTA, A. Imposing essential boundary conditions in meshfree methods. *Comput. Methods Appl. Mech. Engrg. 193*, 12-14 (2004), 1257–1275.

[61] FERZIGER, J. H., AND PERIĆ, M. *Computational methods for fluid dynamics*, revised ed. Springer-Verlag, Berlin, 1999.

[62] FISCHER, H., AND KAUL, H. *Mathematik für Physiker*. Teubner, 1990.

[63] FLETCHER, C. A. J. *Computational techniques for fluid dynamics. 1*, second ed. Springer Series in Computational Physics. Springer-Verlag, Berlin, 1991. Fundamental and general techniques.

[64] FLETCHER, C. A. J. *Computational techniques for fluid dynamics. 2*, second ed. Springer Series in Computational Physics. Springer-Verlag, Berlin, 1991. Specific techniques for different flow categories.

[65] FRANKE, C. Convergence orders of meshless collocation methods using radial basis functions. *Advances in Computational Mathematics 8* (1998), 381–399.

[66] FRANKE, C., AND SCHABACK, R. Solving partial differential equations by collocation using radial basis functions. *Advances in Computational Mathematics 8* (1998), 73–82.

[67] FRIES, T. P. Classification and overview of meshfree methods. *TU Branshweig* (2002).

[68] FRIES, T. P. *A Stabilized and Coupled Meshfree/Meshbased Method for Fluid-Structure Interaction Problems*. PhD thesis, TU Braunschweig, 2005.

[69] FRIES, T. P., AND BELYTSCHKO, T. The intrinsic XFEM: a method for arbitrary discontinuities without additional unknowns. *International Journal for Numerical Methods in Engineering 68* (2006), 1358–1385.

[70] FÜRST, J., AND SONAR, T. On meshless collocation approximations of conservation laws: preliminary investigations on positive schemes and dissipation models. *ZAMM Z. Angew. Math. Mech. 81*, 6 (2001), 403–415.

[71] GIESL, P., AND WENDLAND, H. Meshless collocation: Error estimates with application to dynamical systems. *SIAM Journal on Numerical Analysis 45* (2007), 1723–1741.

[72] GLOWINSKI, R., PAN, T., AND PERIAUX, J. A fictitious domain method for Dirichlet problems and applications. *Computer Methods in Applied Mechanics and Engineering 111* (1994), 283–303.

[73] GOLUB, G. H., AND VAN LOAN, C. F. *Matrix computations*, second ed., vol. 3 of *Johns Hopkins Series in the Mathematical Sciences*. Johns Hopkins University Press, Baltimore, MD, 1989.

[74] GRAYSON, M. A short note on the evolution of surfaces via mean curvatures. *Journal of Differential Geometry 58* (1989), 555.

[75] GREENSPAN, H. On the growth and stability of cell cultures and solid tumors. *Journal of Theoretical Biology 56* (1976), 229–242.

[76] GRIMSON, M., AND BARKER, G. A continuum model for the growth of bacterial colonies on a surface. *Journal of Physics A: Mathematical and General 26* (1993), 5645–5654.

[77] GU, Y. T., AND LIU, G. R. Meshless techniques for convection dominated problems. *Computational Mechanics 38*, 2 (2006), 171–182.

[78] GÜNTHER, F., AND LIU, W. Implementation of boundary conditions for meshless methods. *Computer Methods in Applied Mechanics and Engineering 163* (1998), 205—230.

[79] HACKBUSCH, W. *Theorie und Numerik elliptischer Differentialgleichungen*, second ed. Teubner Studienbücher Mathematik. [Teubner Mathematical Textbooks]. B. G. Teubner, Stuttgart, 1996.

[80] HALTON, J. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerical Mathematics* (1960), 84–90.

[81] HANSEN, P. C. Regularization tools: a matlab package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms 6* (1994), 1–35.

[82] HEINIG, G. Displacement structure of pseudoinverses. *Linear Algebra and its Applications 197,198* (1994), 623–649.

[83] HEINIG, G. On Hankel-like matrices generated by orthogonal polynomials. Tech. rep., Kuwait University, Dept. of Math. and Comp. Sci., 1994.

[84] HEINIG, G., AND HELLINGER, F. Moore-Penrose inversion of square Toeplitz matrices. *SIAM J. Matrix Anal. Appl. 15*, 2 (1994), 418–450.

[85] HEINRICH, B. *Finite difference methods on irregular networks*, vol. 33 of *Mathematical Research*. Akademie-Verlag, Berlin, 1987. A generalized approach to second order elliptic problems.

[86] HIGHAM, N. Fast solution of Vandermonde-like systems involving orthogonal polynomials. *IMA Journal of Numerical Analysis 8* (1988), 473–486.

[87] HIRT, C., AMSTED, A., AND COOK, J. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of Computational Physics 14* (1974), 227–253.

[88] HO, S., YANG, S., WONG, H., AND NI, G. A meshless collocation method based on radial basis functions and wavelets. *IEEE transactions on magnetics 40*, 2 (2004), 1021–1024.

[89] HON, Y., AND CHEN, W. Boundary knot method for 2d and 3d Helmholtz and convection-diffusion problems with complicated geometry. *International Journal for Numerical Methods in Engineering 56*, 13 (2003), 1931–1948.

[90] HU, H., CHEN, J.-S., AND HU, W. Weighted radial basis collocation method for boundary value problems. *International Journal for Numerical Methods in Engineering 69*, 13 (2007), 2736–2757.

[91] HUANG, H., SAIGAL, S., AND DYKA, C. Moving least-square interpolants in the hybrid particle method. *International Journal for Numerical Methods in Engineering 63*, 4 (2005), 528–547.

[92] HUANG, W., REN, Y., AND RUSSELL, R. D. Moving mesh methods based on moving mesh partial differential equations. *Journal of Computational Physics 113*, 2 (1994), 279–290.

[93] HUERTA, A., BELYTSCHKO, T., FERNÁNDEZ-MÉNDEZ, S., AND RABCZUK, T. *Encyclopedia of Computational Mechanics*. John Wiley and Sons, Ltd., 2004, ch. 10 Meshfree methods, pp. 279–309.

[94] HULBERT, G. Time finite element methods for structural dynamics. *International Journal for Numerical Methods in Engineering 33* (1992), 307–331.

[95] IDELSOHN, S., OÑATE, E., CALVO, N., AND DEL PIN, F. The meshless finite element method. In *Fifth World Congress on Computational Mechanics (WCCM V)* (2002).

[96] ISKE, A. *Multiresolution methods in scattered data modelling*. Springer, 2004.

[97] JENSEN, P. Finite difference techniques for variable grids. *Computers and Structures 2* (1972), 17–29.

[98] JIN, X., LI, G., AND ALURU, N. R. Positivity conditions in meshless collocation methods. *Comput. Methods Appl. Mech. Engrg. 193*, 12-14 (2004), 1171–1202.

[99] JONES, D., KOJOUHAROV, H., LE, D., AND SMITH, H. The Freter model: a simple model of biofilm formation. *Journal of mathematical biology 47*, 2 (2003), 137–152.

[100] JONES, D., AND SLEEMAN, B. *Differential equations and mathematical biology*. CRC Press UK, 2003.

[101] KANSA, E. J. Multiquadrics – a scattered data approximation scheme with applications to computational fluid-dynamics I: surface approximations and partial derivative estimates. *Computers and Mathematics with applications 19* (1990), 127–145.

[102] KANSA, E. J. Multiquadrics – a scattered data approximation scheme with applications to computational fluid-dynamics II: solutions to parabolic, hyperboic and elliptic partial differential equations. *Computers and Mathematics with applications 19* (1990), 147–191.

[103] KIM, D. W., AND KIM, Y. Point collocation methods using the fast moving least-square reproducing kernel approximation. *International Journal for Numerical Methods in Engineering 56*, 10 (2003), 1445–1464.

[104] KIRSHMAN, D. J., AND LIU, F. A gridless boundary condition method for the solution of the Euler equations on embedded cartesian meshes with multigrid. *Journal of Computational Physics 201*, 1 (2004), 119–147.

[105] KÖLKE, A. *Modellierung und Diskretisierung bewegter Diskontinuitäten in randgekoppelten Mehrfeldsystemen*. PhD thesis, TU Braunschweig, 2005.

[106] KRONGAUZ, Y., AND BELYTSCHKO, T. Enforcement of essential boundary conditions in meshless approximations using finite elements. Tech. rep., Departments of Civil and Mechanical Engineering, Northwestern University, 1995.

[107] KRYSL, P., AND BELYTSCHKO, T. An efficient linear-precision partition of unity basis for unstructured meshless methods. *Comm. Numer. Methods Engrg. 16*, 4 (2000), 239–255.

[108] KUNLE, M. Entwicklung und Untersuchung von Moving Least Square Verfahren zur numerischen Simulation hydrodynamischer Gleichungen. *Dissertation, Fakultät für Physik, Eberhard-Karls-Universität zu Tübingen* (2001).

[109] LANCASTER, P., AND ŠALKAUSKAS, K. Surfaces generated by moving least squares methods. *Math. Comp. 37*, 155 (1981), 141–158.

[110] LANCASTER, P., AND ŠALKAUSKAS, K. *Curve and Surface Fitting: An introduction*. Academic Press Inc., London, 1986.

[111] LE VEQUE, R. *Numerical methods for conservation laws*. Birkhäuser, 1992.

[112] LE VEQUE, R., AND LI, Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis 31* (1994), 1019–1044.

[113] LEE, S.-H., AND YOON, Y.-C. Meshfree point collocation method for elasticity and crack problems. *Int. J. Numer. Meth. Engng 61* (2004), 22–48.

[114] LEVIN, D. The approximation power of moving least-squares. *Mathematics of computation 67*, 224 (1998), 1517–1531.

[115] LI, Q., SORIC, J., JARAK, T., AND ATLURI, S. N. A locking-free meshless local Petrov-Galerkin formulation for thick and thin plates. *Journal of Computational Physics 208*, 1 (2005), 116–133.

[116] LI, S., AND LIU, W. Meshfree and particle methods and their applications. *Appl. Mech. Rev. 55*, 1 (2002).

[117] LIEW, K., CHENG, Y., AND KITIPORNCHAI, S. Boundary element-free method (BEFM) and its application to two-dimensional elasticity problems. *International Journal for Numerical Methods in Engineering 65*, 8 (2006), 1310–1332.

[118] LING, L., OPFER, R., AND SCHABACK, R. Results on meshless collocation techniques. Tech. rep., Universität Göttingen, Institut für Numerische und Angewandte Mathematik, 2004.

[119] LISZKA, T. An interpolation method for an irregular net of nodes. *International Journal for Numerical Methods in Engineering 20*, 9 (1984), 1599–1612.

[120] LISZKA, T., DUARTE, C., AND TWORZYDLO, W. hp-meshless cloud method. *Computer Methods in Applied Mechanics and Engineering 139* (1996), 263–288.

[121] LISZKA, T., AND ORKISZ, J. The finite difference method at arbitrary irregular grids and its application in applied mechanics. *Comput. & Structures 11*, 1-2 (1980), 83–95.

[122] LIU, G. R. *Meshfree methods. Moving beyond the Finite Element Method*. CRC Press LLC, 2003.

[123] LIU, G. R., AND GU, Y. T. *An Introduction to Meshfree Methods and their Programming*. Springer, 2005.

[124] LIU, G. R., AND KEE, B. A regularized strong-form meshfree method for adaptive analysis. In *Computational mechanics*, C. Mota Soares, J. Martins, H. Rodrigues, and J. e. Ambrósio, Eds., vol. 6 of *Computational Methods in Applied Sciences*. Springer, 2006, pp. 393–412.

[125] LIU, G. R., WU, Y. L., AND DING, H. Meshfree weak-strong (MWS) form method and its application to incompressible flow problems. *Internat. J. Numer. Methods Fluids 46*, 10 (2004), 1025–1047.

[126] LIU, G. R., ZHANG, J., LI, H., LAM, K., AND KEE, B. Radial point interpolation based finite difference method for mechanics problems. *International Journal for Numerical Methods in Engineering 68* (2006), 728–754.

[127] LIU, W., AND CHEN, Y. Wavelet and multiple scale reproducing kernel methods. *International Journal for Numerical Methods in Engineering 21* (1995), 901–931.

[128] LIU, W., CHEN, Y., CHANG, C., AND BELYTSCHKO, T. Advances in multiple scale kernel particle methods. *Computational Mechanics 18* (1996), 73–111.

[129] LIU, W., CHEN, Y., JUN, S., CHEN, J.-S., BELYTSCHKO, T., PAN, C., URAS, R., AND CHANG, C. Overview and application of the reproducing kernel particle methods. *Archive of Computational Methods in Engineering: State of the Art Reviews 3* (1996), 3–80.

[130] LIU, W., JUN, S., LI, S., ADEE, J., AND BELYTSCHKO, T. Reproducing kernel particle methods for structural dynamics. *International Journal for Numerical Methods in Engineering 38* (1995), 1655–1679.

[131] LIU, W., JUN, S., AND ZHANG, Y. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids 20* (1995), 1081–1106.

[132] LIU, W., LI, S., AND BELYTSCHKO, T. Moving least square reproducing kernel methods: (i) Methodology and Convergence. *Computer Methods in Applied Mechanics and Engineering* (1996).

[133] LIU, X., LIU, G. R., TAI, K., AND LAM, K. Radial point interpolation collocation method (RPICM) for the solution of two-phase flow through porous media. In *Third International Conference on CFD in the Minerals and Process Industries, CSIRO* (Melbourne, Australia, 2003).

[134] LÖHNER, R., SACCO, C., OÑATE, E., AND IDELSOHN, S. A finite point method for compressible flow. *International Journal for Numerical Methods in Engineering 53* (2002), 1765–1779.

[135] LOPATYNSKY, Y. B. *Teorija obshich granichnych zadach (in Russian)*. Kiev, Naukova dumka, 1984.

[136] LORENZ, E. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences 20* (1963), 130–141.

[137] LUCY, L. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal 82*, 12 (1977), 1013–1024.

[138] MAISURADZE, G., THOMPSON, D., WAGNER, A., AND MINKOFF, M. Interpolating moving least-squares methods for fitting potential energy surfaces: Detailed analysis of one-dimensional applications. *Journal of Chemical Physics 119*, 19 (2003), 10002–10014.

[139] MATTHIES, H., AND STRANG, G. The solution of nonlinear finite element equations. *International Journal for Numerical Methods in Engineering 14*, 11 (1979), 1613–1626.

[140] MEISTER, A., AND STRUCKMEIER, J. *Hyperbolic partial differential equations. Theory, numerics and applications*. Vieweg and Sohn, 2002.

[141] MELENK, J., AND BABUŠKA, I. The partition of unity finite element method: basic theory and applications. *Computer Methods in Applied Mechanics and Engineering 139* (1996), 289–314.

[142] MONAGHAN, J. Why particle methods work. *SIAM Journal on Scientific and Statistical Computing 3*, 4 (1982), 422–433.

[143] MONAGHAN, J. An introduction to SPH. *Computer Physics Communications 48* (1988), 89–96.

[144] MORTON, K., AND MAYERS, D. *Numerical solution of partial differential equations*. Cambridge University Press, 2005.

[145] MURRAY, J. *Mathematical Biology*. Springer, 1993.

[146] NAYROLES, B., TOUZOT, G., AND VILLON, P. Generating the finite element method: diffuse approximation and diffuse elements. *Computational Mechanics 10*, 5 (1992), 307–318.

[147] NETUZHYLOV, H. Hyperbolic grid generator. Project work, TU Braunschweig, 2002.

[148] NETUZHYLOV, H. Meshfree collocation solution of Boundary Value Problems via Interpolating Moving Least Squares. *Communications in Numerical Methods in Engineering 22*, 8 (2006), 893–899.

[149] NETUZHYLOV, H. Enforcement of boundary conditions in meshfree methods using Interpolating Moving Least Squares. *Engineering Analysis with Boundary Element Method (EABE), Special Issue on Meshless methods 32*, 6 (2008), 512–516.

[150] NETUZHYLOV, H., AND KÖLKE, A. Perturbation technique leading to truly meshfree methods for boundary conditions enforcement. In *Proceedings of WCCM 7, Los Angeles, USA* (2006).

[151] NETUZHYLOV, H., AND KÖLKE, A. Strong form meshfree method for elasticity and fluid-structure interaction problems using a two-field mixed formulation. In *The third international conference on structural engineering, mechanics and computation, Cape Town, South Africa* (2007).

[152] NETUZHYLOV, H., SONAR, T., AND YOMSATIEANKUL, W. Finite difference operators from Moving Least Squares Interpolation. *Mathematical Modelling and Numerical Analysis 41*, 5 (2007), 959–974.

[153] NETUZHYLOV, H., AND ZILIAN, A. Space-Time Meshfree Collocation Method for implicit tracking of dynamic interfaces by level set equations. In *2nd GACM Colloqium on Computational Mechanics for Young Scientists from Academia and Industry* (2007).

[154] NETUZHYLOV, H., AND ZILIAN, A. Implicit reconstruction of irregular complex domain boundaries via scattered data interpolation. *Journal of Computational and Applied Mathematics* (submitted, 2008).

[155] OÑATE, E. Derivation of stabilized equations for numerical solution of advective-diffusion transport and fluid flow problems. *Computer Methods in Applied Mechanics and Engineering 151* (1998), 233–265.

[156] OÑATE, E., AND IDELSOHN, S. A mesh free finite point method for advective-diffusive transport and fluid flow problems. *Computational Mechanics 21* (1998), 283–292.

[157] OÑATE, E., IDELSOHN, S., ZIENKIEWICZ, C., AND TAYLOR, R. A finite point method in computational mechanics – applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering 139* (1996), 3839–3866.

[158] OÑATE, E., IDELSOHN, S., ZIENKIEWICZ, C., TAYLOR, R., AND SACCO, C. A stabilized finite point method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering 139* (1996), 315–346.

[159] OÑATE, E., PERAZZO, F., AND MIQUEL, J. A finite point method for elasticity problems. *Computers and Structures 79* (2001), 2151–2163.

[160] OÑATE, E., SACCO, C., AND IDELSOHN, S. A finite point method for incompressible flow problems. *Computing and Visualization in Science 2* (2000), 67–75.

[161] OSHER, S., AND FEDKIW, R. *Level set methods and dynamic implicit surfaces*. Springer, 2003.

[162] OSHER, S., AND SETHIAN, J. A. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics 79*, 1 (1988), 12–49.

[163] PATANKAR, S. *Numerical Heat Transfer and Fluid Flow*. New York, New York, 1980.

[164] PAULY, M., KEISER, R., ADAMS, B., DUTRÉ, P., GROSS, M., AND GUIBAS, L. Meshless animation of fracturing solids. *ACM SIGGRAPH* (2005).

[165] PERRONE, N., AND KAO, R. A general finite difference method for arbitrary meshes. *Computers and Structures 5* (1975), 45–58.

[166] PESKIN, C. Flow patterns around heart valves: a numerical method. *Journal of Computational Physics 10* (1972), 252–271.

[167] PESKIN, C. Numerical analysis of blood flow in the heart. *Journal of Computational Physics 25* (1977), 220–252.

[168] PICIOREANU, C., VAN LOOSDRECHT, M., AND HEIJNEN, J. A new combined differential-discrete cellular automaton approach for biofilm modeling: application for growth in gel beads. *John Wiley & Sons, Inc.* (1998).

[169] PRESS, W., TEUKOLSKY, S., AND VETTERLING, W. *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, 1993.

[170] QUARTERONI, A., AND VALLI, A. *Numerical approximation of partial differential equations*, vol. 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1994.

[171] QUINLAN, N., BASA, M., AND LASTIWKA, M. Truncation error in mesh-free particle methods. *International Journal for Numerical Methods in Engineering*, in press (2005).

[172] RABCZUK, T., XIAO, S., AND SAUER, M. Coupling of mesh-free methods with finite elements: basic concepts and test results. *Communications in Numerical Methods in Engineering 22* (2006), 1031–1065.

[173] REICHEL, L. Fast QR decomposition of Vandermonde-like matrices and polynomial least squres approximation. *SIAM J. Matrix Anal. Appl. 12*, 3 (1991), 552–564.

[174] ROBERTS, M. E., AND STEWART, P. S. Modeling Antibiotic Tolerance in Biofilms by Accounting for Nutrient Limitation. *Antimicrob. Agents Chemother. 48*, 1 (2004), 48–52.

[175] SAMET, H. *The design and analysis of spatial data structures.* Addison-Wesley, 1990.

[176] SCHABACK, R. Creating surfaces from scattered data using radial basis functions. *Mathematical Methods for Curves and Surfaces* (1995), 477–496.

[177] SCHABACK, R. Error estimates and condition numbers for radial basis function interpolation. *Advances in Computational Mathematics 3* (1995), 251–254.

[178] SCHÖNAUER, W. Generation of difference and error formulae of arbitrary consistency order on an unstructured grid. *ZAMM 78*, 3 (1998), 1061–1062. Selected papers from the Conference on Computational and Mathematical Methods for Science and Engineering (Alicante, 2002).

[179] SCHWEITZER, M. A. Parallel multilevel partition of unity method for elliptic partial differential equations. In *Lecture Notes in Computational Science and Engineering*, vol. 29. Springer, 2003.

[180] SEIBOLD, B. *M-matrices in meshless finite difference methods*. PhD thesis, University of Kaiserslautern, Department of Mathematics, September 2006.

[181] SETHIAN, J. A. *Level set methods and fast marching methods*. Cambridge University Press, 2001.

[182] SHERRATT, J., AND CHAPLAIN, M. A new mathematical model for avascular tumour growth. *Journal of Mathematical Biology 43* (2001), 291–312.

[183] SLEEMAN, B. Solid tumour growth: a case study in mathematical biology. *Nonlinear Mathematics and Applications* (1996), 237–256.

[184] SONAR, T. Difference operators from interpolating moving least squares and their deviation from optimality. *Mathematical Modelling and Numerical Analysis 39* (2005), 883–908.

[185] SPARROW, C. The Lorenz equations: bifurcations, chaos, and strange attractors. *Appl. Math. Sci. 41* (1982).

[186] SRIDAR, D., AND BALAKRISHNAN, N. An upwind finite difference scheme for meshless solvers. *Journal of Computational Physics 189*, 1 (2003), 1–29.

[187] STOODLEY, L., COSTERTON, J., AND STOODLEY, P. Bacterial biofilms: from the natural environment to infectious diseases. *Journal of Industrial Microbiology 2* (2004), 95–108.

[188] STRAUSS, W. *Partielle Differentialgleichungen. Eine Einführung*. Vieweg and Sohn Verlag, 1992.

[189] STROGATZ, S. H. *Nonlinear dynamics and chaos. With applications to physics, biology, chemistry, and engineering.* Perseus Books Publishing, LLC, 1994.

[190] THOMPSON, J., AND STEWART, H. *Nonlinear dynamics and chaos.* John Wiley and Sons, Ltd., 2002.

[191] TIAN, R., AND YAGAWA, G. Non-matching mesh gluing by meshless interpolation - an alternative to Lagrange multipliers. *International Journal for Numerical Methods in Engineering 71* (2007), 473–503.

[192] TOWLER, B., CUNNINGHAM, A., STOODLEY, P., AND MCKITTRICK, L. A model of fluid-bioflim interaction using a Burger material law. *Biotechnology and Bioengineering 96*, 2 (2007), 259–271.

[193] TREFETHEN, L. N. *Spectral methods in Matlab.* SIAM, 2001.

[194] TREFETHEN, L. N. Numerical analysis. Tech. rep., Oxford University, 2006.

[195] VAN LOOSDRECHT, M., HEIJNEN, J., EBERL, H., KREFT, J., AND PICIOREANU, C. Mathematical modelling of biofilm structures. *Antonie van Leeuwenhoek 81* (2002), 245–256.

[196] VARGA, R. S. *Matrix iterative analysis*, expanded ed., vol. 27 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2000.

[197] VILA, J. On particle weighted methods and smooth particle hydrodynamics. *Mathematical Models and methods in Applied Science 9*, 2 (1999), 161–209.

[198] VON NIDA, M. *Meshfree Methods for the Dynamics of Solids*. Shaker, 2006.

[199] WAGNER, G., AND LIU, W. Application of essential boundary conditions in mesh-free methods: a corrected collocation method. *International Journal for Numerical Methods in Engineering 47* (2000), 1367–1379.

[200] WAN KIM, D., AND LIU, W. K. Maximum principle and convergence analysis for the meshfree point collocation method. *SIAM J. Numer. Anal. 44*, 2 (2006), 515–539.

[201] WENDLAND, H. Piecewise polynomial, positive definite and compactly supportd radial functions of minimal degree. *Advances in Computational Mathematics 4* (1995), 389–396.

[202] WENDLAND, H. Computational aspects of radial basis function approximation. In *Topics in Multivariate Approximation and Interpolation*, K. J. et al, Ed. Elsevier, 2005, pp. 231–256.

[203] WENDLAND, H. Scattered data approximation. In *Cambridge monographs on Applied and Computational Mathematics*. Cambridge University Press, 2005.

[204] WENDLAND, H. On the stability of meshless symmetric collocation for boundary value problems. *BIT Numerical Mathematics 47* (2007), 455–468.

[205] WONG, T., LUK, W., AND HENG, P. Sampling with Hammersley and Halton points. *Journal of Graphics Tools 2* (1997), 9–24.

[206] WRIGGERS, P. *Nichtlineare Finite-Element-Methoden*. Springer, 2001.

[207] WU, C.-K., AND PLESHA, M. Essential boundary condition enforcement in meshless methods: boundary flux collocation method. *International Journal for Numerical Methods in Engineering 53* (2002), 499–514.

[208] YOUNG, D. Iterative methods for solving partial difference equations of elliptic type. *Trans. Amer. Math. Soc. 76* (1954), 92–111.

[209] YOUNG, D. L., CHEN, K. H., AND LEE, C. W. Novel meshless method for solving the potential problems with arbitrary domain. *Journal of Computational Physics 209*, 1 (2005), 290–321.

[210] YOUNG, D. M. *Iterative solution of large linear systems*. Dover Publications Inc., Mineola, NY, 2003. Unabridged republication of the 1971 edition [Academic Press, New York-London, MR 305568].

[211] ZHANG, X., LIU, X., LU, M.-W., AND CHEN, Y. Imposition of essential boundary conditions by displacement constraint equations in meshless methods. *Communications in Numerical Methods in Engineering 17* (2001), 165–178.

[212] ZHANG, X., LIU, X.-H., SONG, K.-Z., AND LU, M.-W. Least-squares collocation meshless method. *International Journal for Numerical Methods in Engineering 51*, 9 (2001), 1089–1100.

[213] ZHANG, X. K., KWON, K.-C., AND YOUN, S.-K. Least-squares meshfree method for incompressible Navier-Stokes problems. *Internat. J. Numer. Methods Fluids 46*, 10 (2004), 263–288.

[214] ZHOU, J., AND LI, M. Solving phase field equations using a meshless method. *Communications in Numerical Methods in Engineering 22*, 11 (2006), 1109–1115.

[215] ZHU, T., ZHANG, J., AND ATLURI, S. A local boundary integral equation (LBIE) method in computational mechanics, and a meshless discretization approach. *Computational Mechanics 21* (1998), 223–235.

[216] ZIENKIEWICZ, C., AND TAYLOR, R. *The finite element method.*, vol. 1, 2. Butterworth-Heinemann, 2000.

[217] ZUPPA, C., AND CARDONA, A. A collocation meshless method based on local optimal point interpolation. *International Journal for Numerical Methods in Engineering 57* (2003), 509–536.

Index

“On n’est jamais content là où on est.”

-- Antoine de Saint-Exupéry

“Caminante, son tus huellas
el camino, y nada más;
caminante, no hay camino,
se hace camino al andar.

Al andar se hace camino,
y al volver la vista atrás
se ve la senda que nunca
se ha de pisar.

Caminante, no hay camino,
sino estelas en la mar. ”

-- Antonio Machado